University of Pannonia

Doctoral Thesis

Extending Software Project Scheduling Problems to Investigate Group Selection Mechanisms

Author:
Péter Harta

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

February 26, 2025

Extending Software Project Scheduling Problems to Investigate Group Selection Mechanisms

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: Péter Harta

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 of Business and Management
Department of Quantitative Methods

Doctor of Philosophy

Extending Software Project Scheduling Problems to Investigate Group Selection Mechanisms

by Peter Harta

In today's rapidly evolving and unpredictable economic landscape, companies increasingly adopt resource-oriented and flexible project management methodologies. This trend is particularly evident in the software industry, where agile, extreme, and hybrid approaches dominate. Nevertheless, traditional project scheduling and resource allocation methods often rely on fixed project structures, overlooking the critical role of interpersonal relationships. As a result, these conventional frameworks fail to model and explain why certain flexible practices and team dynamics are essential to effective project management.

This research incorporates DISC behavioral types theory and Belbin's team roles theory into the flexible software project scheduling problems using the synergy potential between each team member. The proposed model shows how a matrix-based project planning approach can handle behavioral types and team roles through the synergies between employees. Moreover, the proposed model can manage flexible projects and level of different type of skills.

Beyond validating the appropriately hyperparameterized model on test databases, this dissertation evaluates its performance through a multi-case study with two real-life case studies. Extending on the synergy-based software project scheduling, it highlights the significance of central team roles and explores the impact of autonomous team role selection.

The research findings shows the importance of considering behavioral types or team roles, validate the effectiveness of autonomous team role selection and demonstrate the pivotal influence of central team roles on the success of software projects.

Keywords: Software project scheduling problem, Behavioral theories, Synergy network, Autonomous team selection, Central team roles

UNIVERSITÄT VON PANNONIA

Zusammenfassung

Doktoratsschule für Wirtschaft und Management
Abteilung für Quantitative Methoden

Doktor der Philosophie

Erweiterung von Softwareprojektplanungsproblemen zur Untersuchung von Gruppenselektionsmechanismen

von Peter Harta

In der heutigen schnelllebigen Wirtschaft setzen Unternehmen verstärkt auf flexible, ressourcen orientierte Projektmanagement-Methoden. Besonders in der Softwarebranche dominieren agile, extreme und hybride Ansätze. Dennoch beruhen traditionelle Planungsmethoden oft auf starren Strukturen und vernachlässigen zwischenmenschliche Faktoren. Dadurch können sie nicht erklären, warum flexible Praktiken und Teamdynamiken für den Projekterfolg essenziell sind.

Diese Forschung kombiniert die DISC-Verhaltenstypen- und Belbin-Teamrollen-Theorie mit flexibler Software-Projektplanung, indem sie Synergiepotenziale zwischen Teammitgliedern einbezieht. Das vorgestellte matrixbasierte Modell berücksichtigt Verhaltenstypen und Teamrollen bei der Ressourcenallokation und ermöglicht eine flexible Projektsteuerung sowie die Anpassung an unterschiedliche Kompetenzniveaus.

Zur Validierung des hyperparametrisierten Modells werden Testdatenbanken herangezogen. Zudem erfolgt eine Multi-Case-Studie mit zwei realen Projekten, um die praktische Anwendbarkeit zu prüfen. Dabei liegt der Fokus auf der Bedeutung zentraler Teamrollen und den Auswirkungen einer autonomen Teamrollenauswahl.

Die Ergebnisse zeigen, dass Verhaltenstypen und Teamrollen entscheidend für den Projekterfolg sind. Sie bestätigen die Wirksamkeit autonomer Teamrollenauswahl und belegen den Einfluss zentraler Teamrollen auf eine effiziente und erfolgreiche Software-Entwicklung.

Stichworte: SoftwareProjektplanungsproblem, Verhaltenstheorien, SynergieNetzwerk, Autonome Teamauswahl, Zentrale Teamrollen

Acknowledgment

I would like to express my deepest gratitude to my supervisor, Prof. Dr. Kosztyán Zsolt Tibor, for his invaluable guidance, support, and mentorship over all these years. I would also like to thank my co-authors, Nóra Obermayer and István Szalkai for their high level support and guidance. Specially thanks for the support of Marcell Tamás Kurbucz and Gergely Lajos Novák for all the support and guidence they have given.

I am deeply grateful to my parents for their unconditional love, encouragement, and belief in me throughout this process. A special and heartfelt thank you goes to my fiancee, whose endless patience, love, and encouragement have been my greatest source of strength. She has been by my side through every challenge, celebrating my successes and lifting me up during difficult times.

Additionally, I would like to express my gratitude to all the employees of Continental who participated in the case study.

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

Contents

List of Figures

List of Tables

Abbreviations

ACO	**a**nt **c**olony **o**ptimization
AHP	**a**nalytic **h**ierarchy **p**rocess
ANOVA	**an**alysis **o**f **va**riance
AoA	**a**ctivity **on** **a**rrow
AoN	**a**ctivity **on** **n**ode
APM	**a**gile **p**roject **m**anagement
ASD	**a**gile **s**oftware **d**evelopment
CPM	**c**ritical **p**ath **m**ethod
CR	**c**onst**r**aint
DISC	**d**ominance **i**nfluence **s**teadiness **c**onscientiousness
DoE	**d**esign **of** **t**hinking
DSM	**d**esign **s**tructure **m**atrix
GA	**g**enetic **a**lgorithm
HAntCO	**h**ybrid **ant** **c**olony **op**timization
HGA	**h**ybrid **g**enetic **a**lgorithm
HPM	**h**ybrid **p**roject **m**anagement
HR	**h**uman **r**esources
IT	**i**nformation **t**echnology
MBTI	**m**yers **b**riggs **t**ype **i**ndicator
MDD	**m**aturity **d**riven **d**evelopment
MDM	**m**ulti **d**omain **m**atrix
MLC	**m**aturity **l**ife **c**ycle
MMRCPSP	**m**ulti**m**ode **r**esource **c**onstrained **p**roject **s**cheduling **p**roblem
MS-RCPSP	**m**ulti **s**killed **r**esource-**c**onstrained **p**roject **s**cheduling **p**roblem
MPx	emertxe **p**roject **m**anagement (↩)
NMM	**n**elder-**m**ead **m**inimization
NSGA-II	**n**on-dominated **s**orting **g**enetic **a**lgorithm - II
PERT	**p**roject/program **e**valuation and **r**eview **t**echnique
PM	**p**roject **m**anager
PMI	**p**roject **m**anagement **i**nstitute
RCPSP	**r**esource-**c**onstrained **p**roject **s**cheduling **p**roblem
RA	**r**esearch **a**ssumption
RQ	**r**esearch **q**uestion
RT	**r**esearch **t**hesis
SMM	**s**ynergy **m**apping **m**odel
SPE	**s**oftware **p**roject **e**nvironment
SPSP	**s**oftware **p**roject **s**cheduling **p**roblem
SSPSP	**s**ynergy-based **s**oftware **p**roject **s**cheduling **p**roblem
SST	**s**cheduled **s**tart **t**ime
TPC	**t**otal **p**roject **c**ost
TPM	**t**raditional **p**roject **m**anagement
TPS	**t**otal **p**roject **s**core
TPT	**t**otal **p**roject **t**ime
UML	**u**nified **m**odeling **l**anguage
Ux/Ui	**u**ser e**x**perience **u**ser **i**nterface
xPM	e**x**treme **p**roject **m**anagement

Symbols

$(\mathbf{Y})$	synergy domain
$(\mathbf{S})$	skill domain
$(\mathbf{M})$	matching domain
$(\mathbf{A})$	logic domain
$(\mathbf{W})$	skilled works domain
$(\mathbf{O})$	output domain
i, j, k	row index, column index, moving index
m	number of employees
n	number of activities
s, s_h, s_s	number of skills, number of hard skills, number of soft skills
e_i	employee i
s_j	skill j
a_k	activity k
ε	subset of all employees m
$[Y]_{ij}$	synergy value between employee i & j
$[S]_{ij}$	j skill of employee i
$[M]_{ij}$	workload dedicated to activity j of employee i
$[A]_{ij}$	precedence value between activity i & j
$[W]_{ij}$	required skill of j for the completion of task i
$[O]_{ij}$	assigned employee j to task i
$\mathbf{M}^T$	transpose of ($\mathbf{M}$) = ($\mathbf{O}$)
$\overline{Y}_{\varepsilon}$	average synergy value of ε
S_j^{ε}	joint skill j of ε
$[\mathbf{W}]_{a,s}$	required skill s to complete activity a
a_j^{dur}	duration of activity j
$a_{j,k}^{dur,adj}$	duration of activity j considering synergy effect of ε who work on it
a_j^{start}, a_j^{end}	start of activity j, end of activity j
e_i^{salary}	salary of employee i
e_i^{maxw}	maximum dedication of employee i in context of required skills
p_{dur}, p_{cost}	duration of the project, cost of the project
TPT_{nosyn}	total project time neglecting synergy effect
TPT_{syn}	total project time considering synergy effect
TPC_{nosyn}	total project cost neglecting synergy effect
TPC_{syn}	total project cost considering synergy effect
C_c, C_t, C_s	cost constraint, time constraint, score constraint
$\mathbb{R}_0^+$	non-negative real numbers
$\mathbb{N}$	natural numbers
$\nexists$	not existing
$\subseteq$	subset
n_F	length of "tasks and dependencies" part of the chromosome
n_s, n_f	number of supplementary tasks, number of flexible dependencies
n_A	number of assignment ratios
N_c	number of elements of a chromosome vector
M	multi-chromosomes

1 Introduction

1.1 Motivation of the thesis

Today's in Industrial Revolution 4 (4IR), the role of the software development is increasing due to the widespread usability of software and the fact that expensive and strict hardware solutions are often replaced by cheaper and flexible software solutions (Wankhede and Vinodh 2021). As software products gain increasing importance, the complexity of their structure and their related requirements are increasing as well (Persson and Mathiassen 2009). Industry Revolution 5.0 (5IR) may mean that software is no longer made by humans but by machines, but today software products must be made by a collaboration of closely cooperating developers. Due to the high volatility in software industry, a new trends of the early 2000s, the emergence of agile project management approach (Beck et al. 2001) in software development (Meckenstock 2024), which builds on the collaboration inherent in the team, stands out. Autonomous/self-organized teams are considered more advantageous in the agile world (Hoda et al. 2012, Kanski et al. 2023), the cross-functionality of the team is also strongly influence the success of the agilely managed software development projects (Gutierrez et al. 2018, Meier and Kock 2023) as well as the team diversity (Albusays et al. 2021, Guimera et al. 2005), and creating the appropriate team dynamics and team structure.

The importance of software development teams on the project scheduling is also underlined by the fact that a widely studied (Vega-Velázquez et al. 2018) so-called Software Project Scheduling Problems (SPSP), deals specifically with the correct selection of human resources in Software Project Environment (SPE). Here, the abilities/skills of the team members fundamentally determine the result of the software development projects, as well as the time and cost of its completion.

Despite the fact that many improvements have been made in the field of SPSP to better align with the agile approach, 58% of the agilely managed projects have failed or faced challenges (Group 2015, 2021). The reason for unsuccessful projects is primarily the inadequate management support and internal resistance to changes (VersionOne 2024). This leads to the fact that in many places there is still a team structure according to the traditional project management as well as the software project scheduling method with insufficient team formation (Aryanee et al. 2020, Gilal et al. 2016) by incompatible and imbalance behavioral types (Vishnubhotla et al. 2018, Bell et al. 2018b) or missing skill-sets (VersionOne 2024). The situation is complicated by the fact that there is a continuous and high fluctuation in software development teams, which causes potential weakening of these groups due to the change of the team structure and team nature. As a result of turnover, a software project team can constantly change, which induces dynamic changes in capabilities (Schulze and Brusoni 2022).

While the impact of behavioral types or personalities and soft skills (Pant and Baroudi

2008) have been considered during MS-RCPSP (Akbar et al. 2022), the incompatibility or imbalance is ignored. Insufficient team formation can be managed and improved by taking into account the synergy potentials (Muniz and Flamand 2023) between different people. Synergy between employees – which represents the result of the common work – and its relation to performance has been extensively studied in the past (e.g. Larson Jr (2013), Liemhetcharat and Veloso (2012)). It is also have shown that synergy is strongly influenced by personality traits (Pieterse et al. 2018). Despite of that, based on the best of my knowledge, Kosztyán et al. (2022) was the first study that considers synergy between employees during the scheduling, and they propose a new class of SPSP, called Synergy-based Software Project Scheduling Problem (SSPSP). However, one of the greatest challenges is to put the theoretical method into practice by identifying synergistic effects between employees. Although most SPSP methods can only model a fixed project structure with predefined tasks, flexible project planning has already been taken into account during SSPSP, such as in agile and hybrid approaches (Wysocki 2019, Reiff and Schlegel 2022), where not all precedence of development processes is fixed. Since a team's effectiveness is determined by its structure, the capabilities of its members, and their personality diversity (Lee et al. 2015, Zainal et al. 2020), as well as the leadership style (Garousi et al. 2019) aligning these factors is essential for achieving more accurate project scheduling.

Although project scheduling methods have been extended to incorporate human factors, no existing approach integrates these aspects comprehensively and takes also into account the characteristics of team dynamics. Consequently, there is currently no method that simultaneously accounts for diverse behavioral types, varied skill sets, and the synergistic interactions between team members, while evaluating their collective impact on project success. These planning shortcomings can readily contribute to the failure of software development projects, because there is currently no uniform method that can be used to model the effect of a change in team structure on team dynamics or self-organized/autonomous team selection.

1.2 Goal of the thesis

Based on the motivation, the goals of this dissertation are twofold: the first goal of the dissertation is to integrate the already existing project scheduling methods, where the consideration of the impact of the team members' behavioral types and their soft skills. The proposed modification will show how to model a software project, where

- property: flexible dependencies are managed,
- property: synergies and behavioral types / team roles of employees are considered, and
- property: soft skills and hard skills are separated.

The second goal of the dissertation is to use this project scheduling model to investigate the following two aspects:

1. aspect: success of the autonomous team versus directed team
2. aspect: impact of the central team roles on the success of the software projects

Throughout the dissertation, I interpret the project as an external project. In this context, two main actors of the projects can be distinguished: the project owner organization that initiated the project (hereinafter referred to as project owner or customer), and the project-based organization who perform the project. I interpreted my research from the point of view of the organization performing the project, i.e. I deal with the investigation of the performance of the ordered project.

The validation of the new method in a real software development environment, based on case studies, can be defined as a side goal of the dissertation.

1.3 Research questions

The following research questions will be answered:

RQ$_1$ How can a software project scheduling method be enhanced to incorporate the uniqueness of different team roles and behavioral types, while also considering their interactions within a heterogeneous network, shaped by diverse skill sets and synergy effects, in both structured and flexible environments?

RQ$_2$ How do central team roles as the central unit of a heterogeneous network influence the success of software projects through their integration into scheduling strategies?

RQ$_3$ How does autonomously selected team as a heterogeneous network affect the success of software projects through their scheduling?

1.4 Structure of the thesis

The further chapters of the thesis are structured as follows: Section 2 gives an overview about the most relevant articles. Section 3 introduces the proposed model. Section 5 presents the case studies, where I show how to fill the data tables required for the new method with data from real companies' environment. Section 6 presents the results of the simulations. Section 7 defines the method of validating the simulation's results in real environments and Section 8 presents the results of the validation with the discussion of the interpretation of the results and finally, Section 9 concludes the thesis.

2 Literature review

This chapter is designed to facilitate a deeper understanding of the research component. The dissertation focuses on the scheduling of software projects and the selection of project teams especially to summarize the relevant research so far and shed light on the gaps. Therefore, this chapter includes three major topics: projects and project management, software project scheduling and team selection methods. Both chapters place special emphasis on the description of the industrial characteristics of the software.

2.1 Project and project management

2.1.1 Interpretation of the project

The conceptual interpretation of the project has undergone a significant evolutionary change in recent years (Wawak and Woźniak 2020), but there is no conventional definition describing the interpretation of the project. Just as the interpretation of the project is also an evolutionary process (Shenhar and Dvir 2007), the concepts of the project created in different periods can also be different.

The project is originally a concept as old as human society, but its significance in the life of organizations only appeared in the 20th century. At this time the project as a process was defined, which is a sequence of unique, complex, and connected activities that must be completed by a specific time, within budget, and according to specification (Wysocki 2019). This project interpretation (Olsen (1971)), which can be delimited by the project triangle (De Wit 1988) is also interpreted in operations research, especially in the context of project scheduling, focusing on its uniqueness (Görög and Smith 1999), complexity and constraints (Kosztyán 2016).

The starting point was the connection of the conceptual interpretation of the project with the organizational strategy, which created the concept of a strategy-oriented project at the end of the 20th century. This new direction essentially comes from the work of (Cleland 1994), who defined a project as a set of tasks that an organization must perform with specific goals to be a means of implementing the strategy. This interpretation of the project considers projects as part of the corporate strategy (Cleland 1994, Görög 2003), as a kind of implementation tool, beyond the definition of project as a process. In the corresponding literature, projects can be grouped by many of their properties like indicated changes in the company (Wheelwright 1992), results (Wateridge 1999) or as a building blocks of strategy (Cleland 1994, Görög 2007, Görög and Smith 1999).

In addition to defining the project as a building block of the strategy, a new point of view that appeared almost in the same period discusses projects as temporary organizations. That view of the project as a temporary organization (Lundin and Söderholm 1995, Söderlund 2004, Turner et al. 2010) underscores the importance of the team. However, this presupposes teams unique to the project, but they cease to exist at the end of the project and give way to new teams of new projects. In the context of project as a temporary organization, the definition is given by Turner et al. (2010), who emphasizes the importance of unique, novelly organized teams with their own resources who create the project result under time and cost constraints.

However today's, thanks to the dynamic approach of the projects (Beck et al. 2001, Winter 2015), these are not only considered as the changes embedded in the strategic goals are implemented or as a temporary organization, rather, it is understood as a combination of these in a holistic way (Bredillet 2008, Shenhar and Dvir 2007).

As today's, in the middle of the Industrial Revolution 4.0 the importance of software products are increasing, therefore in this dissertation it has become more important to study projects that deal with the creation of software. These are the so-called software projects which involves people effort, organized in teams, working together in an environment that is in constant change, with unstable project parameters (Chang et al. 2008). These software development projects belong to the group of Information Technology (IT) projects and aim to create a unique software. (Wysocki 2010) defined the software development projects as a complex venture by at least two persons which is delimited by time, budget and human resources to create new or enhanced computer code for a purpose of a new and significant business value or process.

For a better understanding of the definition and evolution of (software) project, I describe the conceptualization of the variables – especially for the software projects – included in it in the following subsections:

- project cycle to explain the "project as a process" conception
- project success to explain the part of the definition: "within the boundaries of time, budget, and staff resources"
- project management definitions and
- project management approaches

2.1.2 Evolution of the project cycle

Since the projects were organized to achieve a unique goal, projects as unique processes can generally be broken down into stages. The so-called project cycle is used to represent the project in general as a process, which according to Görög (2003) a conceptual framework of the feasibility of the project, which should reflect projects in organizations also played its strategic role. Although the development of the interpretation of the project definition now clearly emphasizes the strategic orientation, the integration of the project's life cycle into the organizational strategy is still ignored by most authors. However, this way of thinking appears in the work of Görög (2003), in which the most important or critical decision points that separate the individual project phases are the first to appear. This strategy-oriented project cycle illustrates the key phases of a project through an iterative approach, whereas other authors, such as Corsten and Corsten (2000) and Cleland (1994), interpret the project cycle as a linear process. Essentially, both approaches divide the project process into 4 main parts:

Table 1: Sequence of the project cycle

-	Cleland (1994)	Corsten and Corsten (2000)	Görög (2003)
Phase 1	Project concept	Project definition	Project design
Phase 2	Project design	Project design	Awarding
Phase 3	Project execution	Project execution	Project execution
Phase 4	Project end	Project release	Post-analysis

The project usually begins with the birth of an idea, while in the case of a strategy-oriented project cycle, the idea can be derived from the corporate strategy, so its first phase already provides an opportunity to evaluate the different project versions and introduce the best project version. Furthermore, during the progress from the first stage to the second stage, the feasibility conditions of the project determining the success of the project must be recorded in the strategy-oriented project cycle model. The second phase of the project's cycle is about project planning, while the Görög (2003) model provides an opportunity to interpret the external contributors and their competition in the conceptual framework. In the third phase of the project cycle, the achievement of the project's results takes place within the predefined framework, which in the case of the strategy-oriented project cycle approach is preceded by the responsibility and risk assessment defined in the project contract as the second decision point. Finally, the final phase can be defined as closing the project or handing over the project results. It is important that in the case of the strategy-oriented approach, the project is finished, but turns into a lessons learning phase, which is allowed by the acceptance of the project result as the third decision point. While the evaluation of the project as a whole can be observed in all three examined cases simultaneously with the closing of the project, the Görög (2003) model defines this evaluation as part of the organizational strategy, which can be used to understand the organizational innovation indicated by the projects. Although the conceptual framework of the project has changed over the years with economic development, the main steps of the project cycle model have not. On the other hand, the structural change that occurred in the steps can be observed, which I will present on the example of software projects.

The project cycle for software projects has a more specialized structure, but essentially follows the life cycle model for projects in general. Based on Wysocki (2010) the software project cycle contains the following tasks: (1) Requirements Gathering, (2) System Design,

(3) Detailed Design, (4) Code and Test, (5) Systems Test. Software project management in the traditional sense represents this linearly, using the so-called waterfall model (Figure 1)

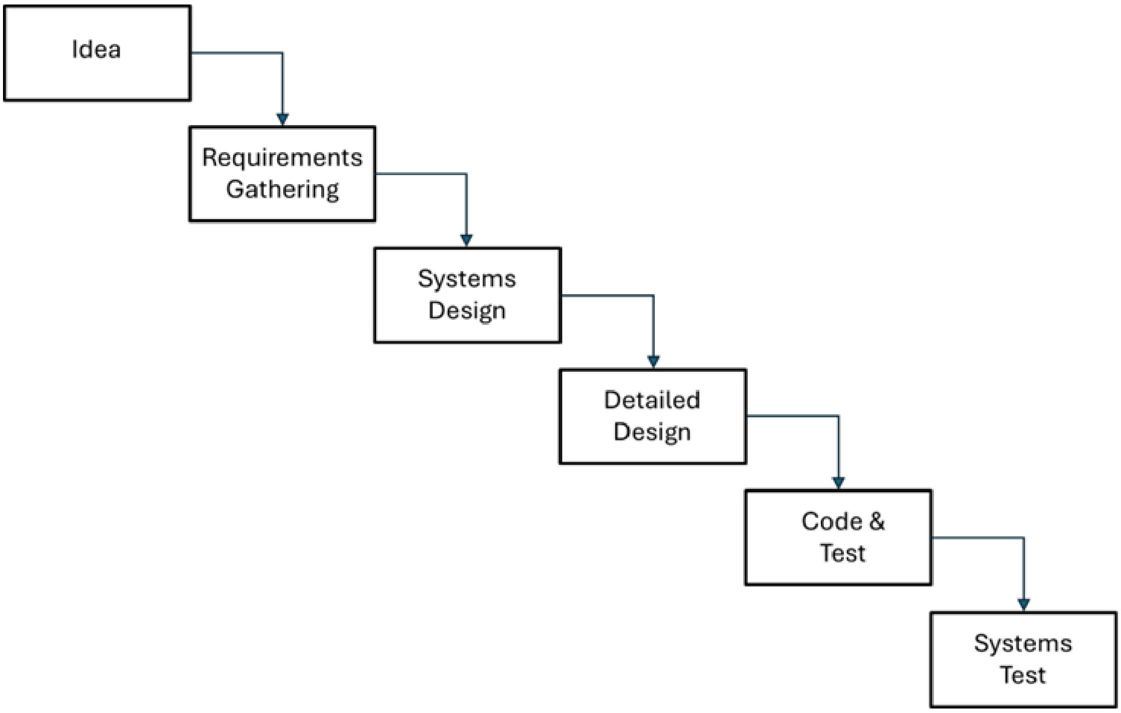


Figure 1: Standard Waterfall Model

In software development, often in the case of larger projects, the needs of the project owner change as the project progresses. Therefore, they often do not know exactly what kind of project result they will be satisfied with, they only have a basic idea. The so-called design thinking (Plattner et al. 2009) supports the project-based organizations in giving project owners a prototype product before releasing the final project result, and then developing the existing prototypes according to the needs of the project owner.

In practice, it often happens that the owner of the project needs one or more prototype products that are already more mature based on their requirements before receiving the final product, so they can even start their own tests with a rudimentary product. In such cases, it is advisable for the project-based organization to deliver the increasingly mature product to the owner of the project in smaller units, so-called increments. This is described by the Staged Delivery Waterfall model (Wysocki 2010), which is a kind of response of the project management literature to the emergence of customer-oriented projects (Figure 2).

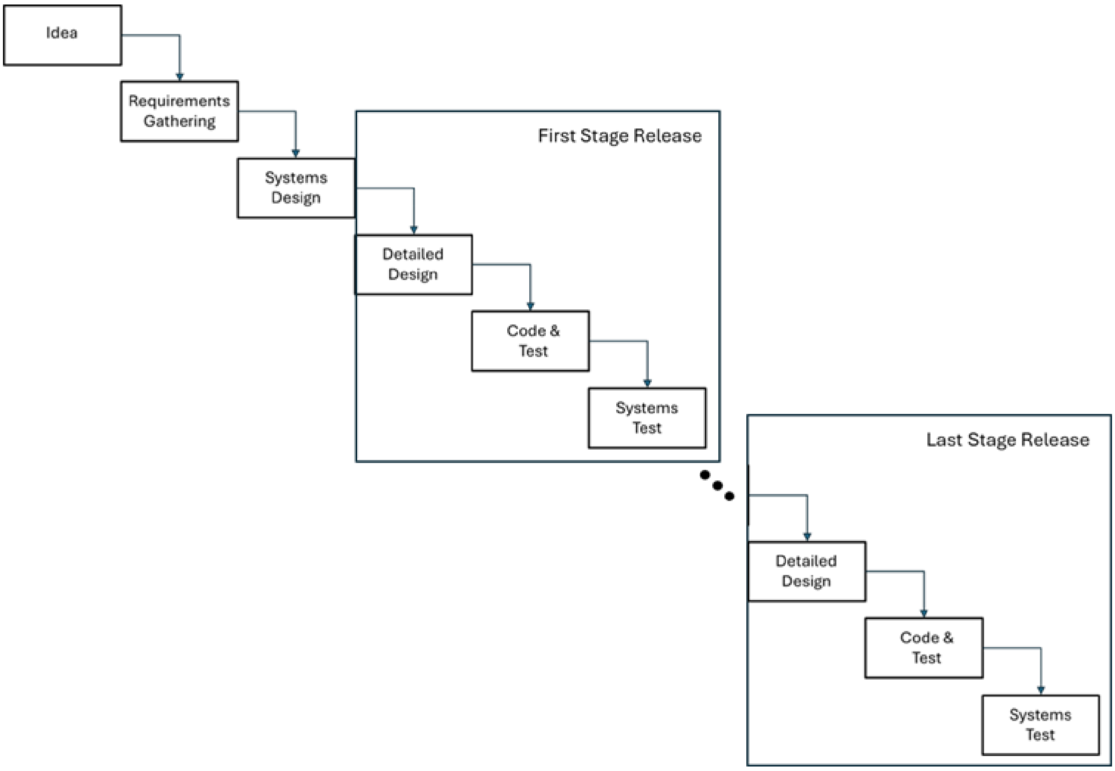


Figure 2: Staged Delivery Waterfall Model

However, in today's fast-changing and dynamic world, it is not only necessary to increase the efficiency of development and testing with incremental development, but requirements from the project owner may also change during the life of the project in accordance with market needs. It is worthwhile to deliver the increments at such certain intervals in order for the project-based organization to provide the project owner with the opportunity to reconsider their requirements. This software development process is described in the Evolutionary Development Waterfall model (Wysocki 2010), which is a characteristic of the flexible (sometimes agile) project management literature, which later became the basis of the hybrid approach (Figure 3).

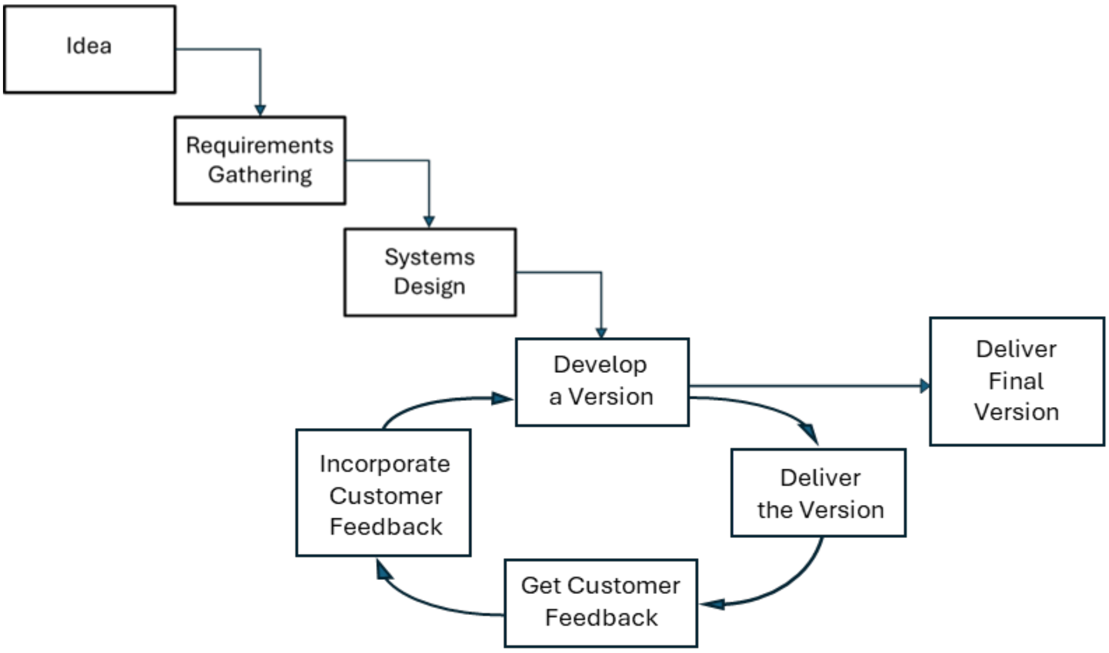


Figure 3: Evolutionary Development Waterfall Model

The structural change of the evolutionary cycles thus followed the conceptual change of the projects over the years and facilitated the successful implementation of the projects. Therefore, the way of completing projects, which is the responsibility of project management, has also changed.

2.1.3 Project management

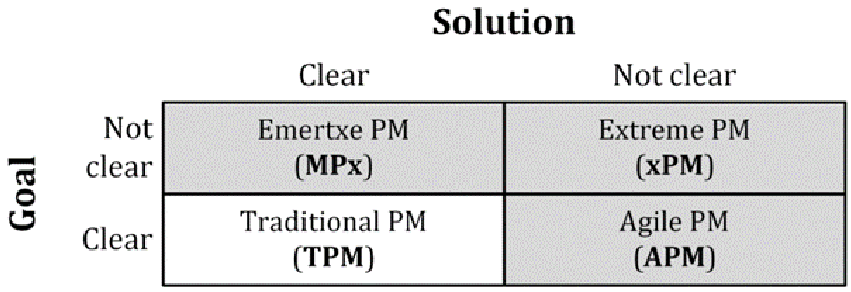
Since projects are complex processes, they must be managed in terms of their successful implementation. Project management is responsible for ensuring this. Project management is primary responsible to schedule, monitor and control the projects through their whole life cycle (Phillips 2018). It can be concluded that with the development of the interpretation of projects, the interpretation of project management has also changed. Görög and Smith (1999) focuses on the organization level of project management as a separate management activity that ensures the completion of complex tasks and lead the project team. In today's world, organizations basically organize their activities in projects. Because of this, projects, or at least a part of them, have been outsourced to external contributors, especially in the case of larger information projects. In this context, we distinguish between a project owner organization, which initiates the project, and a project-based organization, which implements the project result - expected by the project owner organization - as an external contributor. Since the effectiveness of the organizations is significantly determined by the success of the projects to be completed at the same time, these projects were organized into the organization's project portfolio.

In addition, software projects are often complex like other projects. These require effective organization and planning, which are provided by project management. Project management should plan the tasks and their resources; organize the assignment between peoples, tasks, and resources; leading teams and peoples; controlling the deviations to achieve the set goals (Görög 2003). The definition of software project management is given by (Wysocki 2010), according to whom the software development project management is a unique discipline of evaluating the specifics of the software to be developed, choosing the appropriate project management approach and best fit software development life cycle to guarantee that the project owner's needs are met in delivering business value with maximum effectiveness and efficiency.

Since projects are different, different project management approaches have been created and can be applied more successfully in individual areas (Salameh 2014). Therefore, it is worth taking a closer look at these project management approaches.

2.1.4 Project management approaches

Managing projects (and also software projects) can be approached in several ways. Wysocki (2019) separated the project management approaches into four fundamental areas regarding the dimensions of their goals and solutions (Figure 4). He distinguishes Traditional Project Management (TPM), Agile Project Management (APM), Extreme Project Managament (xPM) and Emertxe Project Management (MPx). In the practical life the TPM and the APM are widespread while MPx and xPM are mostly related to the research and education field (Toljaga-Nikolic et al. 2017).



		Solution	
		Clear	Not clear
Goal	Not clear	Emertxe PM **(MPx)**	Extreme PM **(xPM)**
	Clear	Traditional PM **(TPM)**	Agile PM **(APM)**

Figure 4: Project Management Approaches

While from the 50's after the 2nd World War to the end of the 20th century when the traditional approach was founded the goal and the solution were clear. Industry focused mainly on manufacturing where mass production of mature constructions took place and raw materials were inexhaustible and cheap. In the context of traditional project management, project owners know what they want, and the project-based organizations have the proper solution to serve project owners' requests. TPM consists of well-defined and functional project management process groups that are connected to each other through their outputs. The most popular process model of traditional project management is the Waterfall Model (see Section 2.1.2) that operates in a fix and sequential manner (Binder et al. 2014) and derives from the linear structure. Coming to the end of the 20th century the price of raw materials continued to rise, along with the popularity of software, while continuous innovation and cost reduction trends were increased. Production faced constantly changing cheaper substitute products instead of clear requirements about the product and software products has become increasingly important, because they do not need expensive parts. Project owners always wanted to have the newer and better solutions then the software industry became turbulent and uncertain (Conforto et al. 2014). Software projects – which had been used traditional approaches – are characterized by large cost and time overruns, as well as the use of additional resources. Meanwhile, conflicts in the software teams occurred due to the lack of the communication and the continuous changes of the software content (Dybå et al. 2014). Recognized the uncertainty and complexity in software development area, the Agile Project Management (APM) was founded to act proactively in a constantly changing environment.

Basics of APM was originally written down in the Agile Manifesto (Beck et al. 2001), but the first analogy of agility had been mentioned earlier by Takeuchi and Nonaka (1986), where the authors talked about it as the quick response to changes and called it to "scrum" from the rugby. Agile project management is defined by short cycles of iterative and incremental delivery of product features, combined with the continuous integration of code changes. APM adopts a feature-driven approach, emphasizing the prioritization of project features and requirements based on their score value. Consequently, active customer participation in defining the project scope and analyzing its requirements is essential. With the help of good prioritization, the work of the project-based organization can be supported and the satisfaction of the project owner organization can be increased.

APM prescribes the delivery in increments iteratively like the Dynamic Systems Development Method (Stapleton 1997) and frequent software release cycles like to the Extreme Programming (Beck 1999). After the Agile Manifesto several agile methods were created. According to Salo and Abrahamsson (2008), Meckenstock (2024) Scrum method is the most common and popular method among the agile methods. Many Scrum aspects are common to agile methodologies, such as iterations, incremental development, self-organized teams, and flexibility in the face of changing requirements (Schwaber and Beedle 2001, Gupta et al. 2022). Therefore, scrum is sometimes identified as the agile methodology.

Scrum prescribes to separate the development tasks – called product backlog – into small parts, called sprints. Sprints spend 2-4 weeks, starts with the definition of sprint backlog, and ends with the evaluation of the sprint review. To ensure the continuous learning and innovation scrum has a project control function, the sprint retrospective where all experiences from a sprint are summarized and evaluated. The development depends on the self-organized team driven by the Scrum Master. While the Scrum Master enforces the principles of scrum and holds the daily scrum in the role of the development team leader, the Product Owner manages the product development with his prioritization. In the software industry particularly for the scrum – which is more team-based than the other methods – the “pull principle” is important to plan the workload based on the team members (Kniberg and Skarin 2010). Pull principle is originally described in the kanban method but to implement this principle into the scrum, they are often used together as Scrumban (Corona et al. 2013).

However, agile methods are increasing traditional methods are still used today in the software industry (Wysocki 2010). Fulfillment of software projects can be differentiated when traditional or agile approach is used to manage the projects. With traditional approach the scope of the project is sometimes well-known and predefined by the priority of the project tasks, while the project cost and the project duration (time) can be changeable to achieve the original scope. In constraints agile approach supports to change the priority of project tasks when the project owner requires it to be able to provide the fast reaction for the continuous changes (Figure 5). In the case of projects managed with agile methodologies, planning receives a much greater focus, support flexibility and autonomous team and iterative progress (Dybå and Dingsøyr 2008, Highsmith and Cockburn 2001).

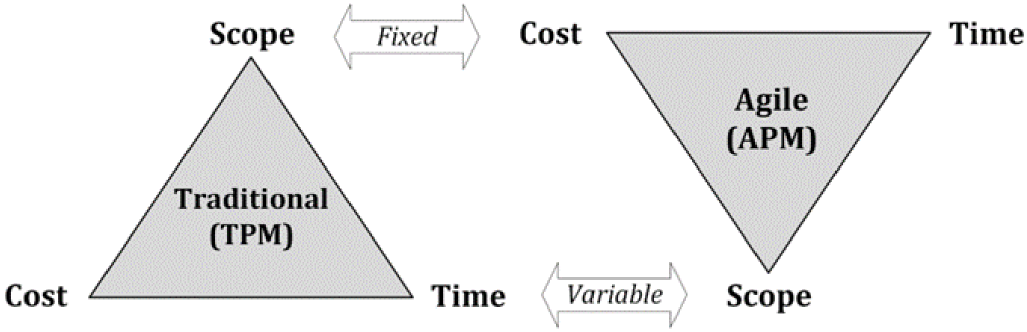


Figure 5: Difference between the TPM and the APM based on the constraints

Beside of the separation of traditional and agile projects, hybrid methods, like waterfall-agile, waterfall-scrum, hybrid-V-model or Agile-Stage-Gate increased considering the advantages from both methods (Reiff and Schlegel 2022). But disadvantages should also be considered because the hybrid methods require much more preparedness. Basic differences between traditional, agile and hybrid methods are shown in Table 2.

Table 2: Comparison of Traditional, Agile and Hybrid project management approaches

Trait	Traditional	Agile	Hybrid
Basic assumption	All software related requirements are clear at the planning phase	Software is developed by small teams using the principles of continuous design improvement	Mix the advantages of traditional and agile approaches, maximization of project success
Management style	Command and control	Leadership and collaboration	Customer-centric approaches
Project structure	Fix	Flexible	Flexible
Constraints	Fix	Fix	Optional
Project control	Hierarchic, pre-defined control	Subtle control, team-focused control	Matrix-based control
Knowledge management	Explicit	Tacit	Comprehensive methodology
Communication	Formal	Informal	Transparent and complex, flexible response to changes
Development model	Life-cycle model	The evolutionary-delivery model	Both traditional and agile models
Desired organizational structure	Large and highly structured	Small and medium-sized organizations with high cooperation and flexibility	High number of team members, well networked, open to new methods
Quality control	Heavy planning and testing, strict control	Continuous control and testing, encourage change and constant feedback	Continuous control and testing, where iterative part is used. Increased administrative effort
Result of the project	Rare (often one) delivery based on all requirements	Frequent delivery (increments) based on separated and prioritized requirements	Rare or frequent delivery to the project owner but frequent in the organization (continuous testing)
Execution plans of projects	Project life cycle model	Evolutionary, iterative model	Mixed
Cooperation with the project owner	Low, by contract	High, involved in the work	Depends on the method but it is often high
Requirements	Well-known and well-defined	Continuously changing	Pre-defined but partially changeable
Best project's size	Large	Small or medium	Based on the complexity
Priority value	High safety	Fast delivered value	Mixed

Considering both positive and negative effects of each approach it can be recognized that somewhere the traditional elsewhere the agile or the hybrid approaches can be expedient. Wysocki (2019) found that in practice, in the vast majority of cases, the hybrid solution is used, in which both traditional and agile approaches can be recognized. A purely traditional or agile approach, which was much more common in the past, is now rarely implemented.

With a properly assessed project portfolio and the appropriately selected project management approach, the degree of its successful execution directly influences the long-term success of project-oriented organizations. For this reason, it is worthwhile to deal with the concept of project success in more detail.

2.1.5 Success of projects

Based on the definition of the projects in context of project as a process, a (software) project is successful if, during its completion, the predefined (software) project result is achieved within a defined cost, time. This conception of project's success constraints is the so-called iron triangle or project triangle (De Wit 1988), in which the previous experts combined the 3 main criteria (project result, project cost and project duration) measuring the effectiveness of the project and named them the project's success criteria (Cooke-Davies 2002).

The change in the concept of the project, as a means of implementing the strategy, opened the way for the intellectual development of the success of the project. In this context, the satisfaction of the project owner organization implementing the project also became important, i.e. the extent to which the project fulfilled the goals included in the organizational strategy (Atkinson 1999, Baccarini 1999). However, this did not override the success criteria defined by the iron triangle, which measured the effectiveness of the project organization, but rather introduced the efficacy of the project organization. Therefore in this context, the project can be successfully implemented if it fulfills the company's strategy, which is why the project was created and it does not exceed the limits defined through the various success criteria. The evolution and emergence of the project's success is strengthened by Davis (2017) and Koops et al. (2016), who, in addition to the success criteria defined within the framework of the project triangle, consider satisfaction of the project owner as a success criteria. A more recent study (Pankratz and Basten 2018) already puts quality in the background compared to satisfaction of the project owner, in addition to the factors of time and cost.

Beyond that, the further development of the interpretation of the project - as the project is a temporary organization - induced the further development of the success criteria. In this context, the project was created not only for the implementation of the strategy, but also defined a separate team. In this context, the efficacy of the project extended to the satisfaction of the project stakeholders (Lundin and Söderholm 1995, Jugdev and Müller 2005). From 2015 the yearly Chaos Report by the Standish Group changed its evaluation to define the success of (software) projects to *"the project was resolved within a reasonable estimated time, stayed within budget, and delivered customer and user satisfaction regardless of the original scope"* (Group 2015).

Although the interpretation of project success has evolved, the different success criteria are not separate, but can be arranged in a hierarchy. The Hierarchical Criteria Model (Figure 6) was presented by Görög (2007) who categorized the projects' success criteria into 3 level. The large-scale changes in the flexibility requirements of the projects induced flexible approaches such as agile planning, the effects of which are also reflected in the critical success criteria (Wawak and Woźniak 2020). In a recent study, Binboga and Gumussoy (2024) united the efficacy levels ("satisfaction levels") of the hierarchical model and, placing a special focus on sustainability, divided the agile success criteria into the following 3 groups:

- process efficiency: representing the pillars of the iron triangle
- sustainable software product quality: e.g., quality, reliability, compatibility, sustainability, usability
- stakeholder satisfaction: project owner, team and management satisfaction

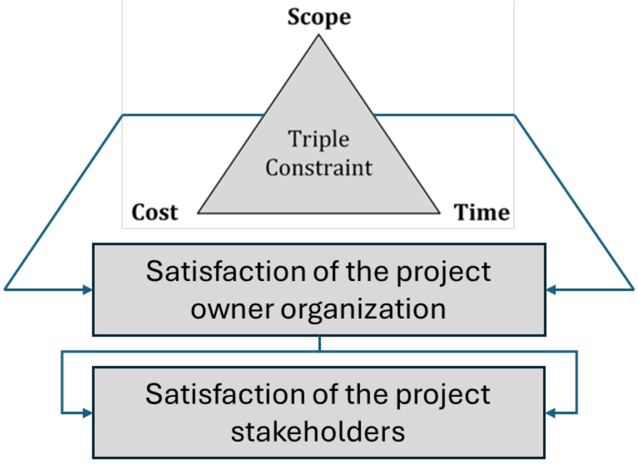

Scope
Triple Constraint
Cost
Time
Satisfaction of the project owner organization
Satisfaction of the project stakeholders

Figure 6: The Hierarchic Model of the project success criteria: Own illustration based on Görög (2007)

The interpretation of the project's success factors also changed along with the development of the conceptual interpretation of the project. Distinguished from the success criteria, success factors are circumstances that shape the degree of success (Cooke-Davies 2002, Bredillet 2008).

Previously based on the work of (De Wit 1988) and (Pinto and Slevin 1988) the success factors of the project included the professionalism of the project manager, the planning effort, the political environment, the understanding of the project mission, cooperation with the stakeholders, and the correct interpretation of the importance of the project. After that, 12 project success factors by (Cooke-Davies 2002) were born in the spirit of a strategy-oriented project organization and appear as success factors such as cooperation between line management and project management, the effective incorporation of strategy into projects, or the effectiveness of learning from experience.

After the emergence of the conceptual interpretation of the project as a temporary organization, the importance of stakeholders began to play a central role (especially in the world of software development) and this category of success factors became of primary importance. However, as a consequence of the formalization and separation of project types, the success factors are now much more specific to the project type than can be easily generalized.

By analyzing the literature, Iriarte and Bayona (2020) identified the 4 most important success factors in software development in the context of project as a temporary organization. These are the involvement mainly from the users; support, mainly from the top management; communication, mainly in house; and knowledge and technical expertise, mainly from the consultants

It is important to note that among the success factors there are few examples regarding the team or team members, and those also highlight the skills and capabilities of the team members. In software environment, a particularly important success factors, the existence of the resource, needs to be mentioned here. In software environment, it is mostly human resources that determine the success of a software project (Sudhakar 2016), and the human-related factors can also be separated into 3 main parts based on Misra et al. (2006):

- technical factors: e.g., requirements, development process, testing
- organizational factors: e.g., team structure and size, organizational culture
- individual factors: competencies, personality traits, communication, learning

The project program, project success factors or risk analysis, which were defined as the central concepts of project management interpretations in the early 2000s, are now built around project management maturity, mega projects, agile portfolio, disaster recovery or even sustainability (Wawak and Woźniak 2020). Today's several groups of success factors are separated, the focus of which is sustainability and the human factor. But only recently created articles treat the project team as a separate category among the success factors. According to Binboga and Gumussoy (2024), they can be classified into the: organizational factors; team factors; customer factors; technical factors; agile process factors; and project factors.

As the success criteria and success factors developed and became more and more aligned with the agile approach, their widespread application in industry also gradually took place. Based on the yearly Chaos Report by Standish Group the success rate of the agile projects is increasing more against the traditional projects. Basis of the comparison is that the success of the agile projects can be measured in the same way as traditional projects using the definition related to the project's success by Group (2015).

Table 3: Success of projects - Chaos Reports

	Group (2015)		Group (2021)	
	Traditional	Agile	Traditional	Agile
Successful	11%	39%	13%	42%
Challenged	60%	52%	59%	47%
Failed	29%	9%	28%	11%

Within the evaluation of different project management approaches, the evaluation of the agile approach to software development is done by the yearly agile survey by VersionOne. They said that 61 % of the respondents reported that" most" or "all" of their agile projects have been successful. Adopting agile can accelerate software delivery, increase project visibility, and team productivity (VersionOne 2017). However, these survey's results were higher in VersionOne (2013), it can be seen in Table 4.

Table 4: Success of projects - Reports of VersionOne

Trait	VersionOne, 2013	VersionOne, 2017
Manage changing priority	90%	71%
Increase software's productivity	85%	55%
Project visibility	84%	66%

The comparison suggests that the efficiency of agile methods is decreased however, based on 98 % of the respondents had success with agile projects, can be seen in VersionOne (2017). According to the recent survey by VersionOne (VersionOne 2024), agility is gaining more and more attention, but the industry is finding it difficult to adapt to this new approach. According to their survey, this can be traced back to the general resistance to organizational change or culture clash, lack of the leadership participation or inadequate management support. This can be caused by the fact that in the case of companies (especially large companies) too many processes are still tied to the traditional approach, and that project teams still form so-called silo groups instead of cross-functional groups. Although nowadays in software development projects are most often managed according to the agile approach, different hybrid approaches are receiving more and more attention (VersionOne 2024).

In order for the success rate of projects to be high, it is not enough to apply the appropriate project management approach, but it is also necessary to have an appropriate project schedule carried out by project management. In the absence of an improperly designed schedule, it is extremely difficult to meet the cost and time constraints in the case of complex projects, like the software projects.

2.2 Project scheduling

With the evolution of the interpretation of projects, of course, the method of planning and scheduling projects also changed. Focusing on the much broader project scheduling, it can be defined as follows:

"Project scheduling is the application of skills, techniques, and intuition acquired through knowledge and experience to develop effective schedule models. The schedule model integrates and logically organizes various project components, such as activities, resources, and logical relationships, to enhance the likelihood of successful project completion within the baseline duration." (PMI 2017)

To present this schedule a schedule model can be used, which is defined as follows:

"Schedule model is a dynamic representation of the plan for executing the project activities developed by the project stakeholders, applying a selected scheduling method to a scheduling tool using project-specific data." (PMI 2017)

Schedule model is generated by using a schedule method with a scheduling tool and the related information comes from the project itself (PMI 2017).

During the project life cycle all specific unit of the project, called task or activity must be scheduled and performed. A task (or activity) is a "distinct, scheduled portion of work performed during the course of a project" (PMI 2017). The tasks are specific to the types of projects. Although all software projects are unique, there is a general work breakdown structure which describes what their most important tasks are. (Figure 7).

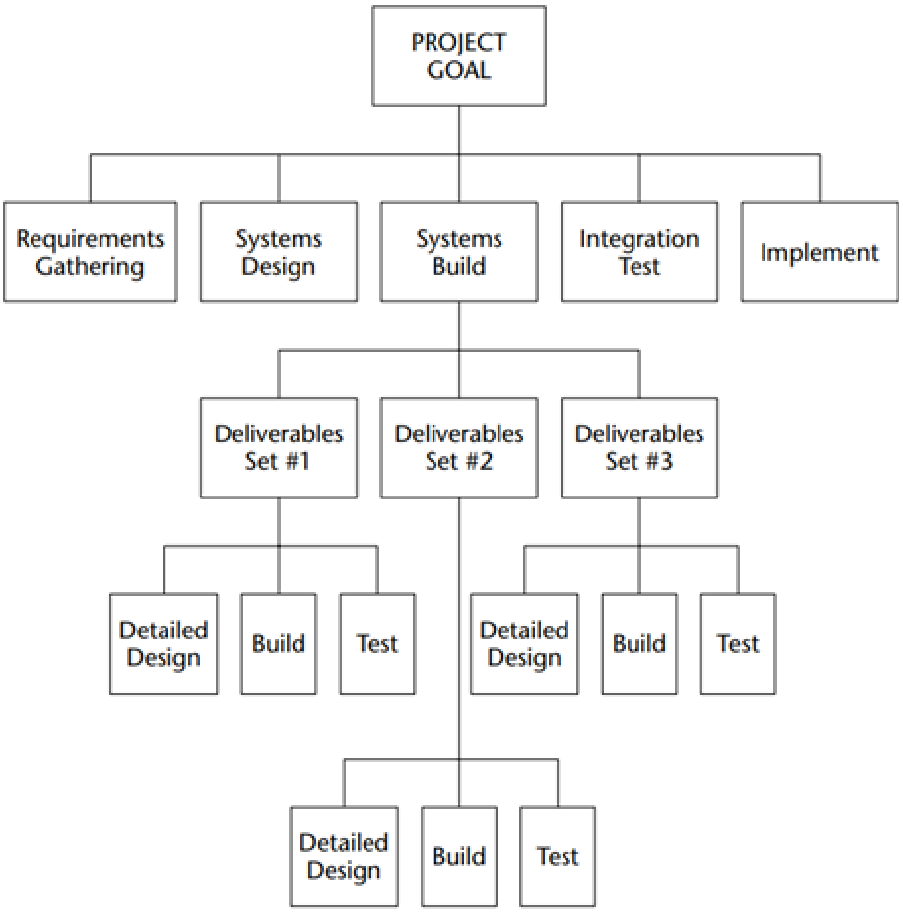


Figure 7: General Software Work Breakdown Structure

All tasks require a start and an end event and, in most cases, previous (called predecessor) and subsequent (called successor) activity or activities. The relation network between the tasks is represented by the precedence graph or precedence matrix. Regarding the network diagrams, two basic notation technical solutions have been developed to display the activities included in the project (Görög 2003):

- activities as arrows, or the so-called NoA (Notation on Arrow) network
- activities as geometric shapes or the so-called NoN (Notation on Node) network

Project scheduling methods proposed from the 1960's with network-based methods (based on AoA or AoN), where the searching of the best total project time with critical path methods (CPM) (Antill and Woodhead 1991, Kelley Jr 1961) or PERT (Van Slyke 1963, Malcolm et al. 1959) was done. CPM uses deterministic and PERT uses stochastic time and cost values. However, it's important to note that these approaches are constrained to identifying the critical path in an unconstrained environment and do not factor in any resource limitations. Moreover, CPM and PERT are widely used network planning methods, due to their rudimentary nature, they cannot be used to manage changes in strategic goals (Kosztyán 2012).

As projects became more intricate and its uncertainty was increased (Pich et al. 2002), especially with the rise of complex software projects, the significance of flexible approaches has grown, leading to the adoption of flexible planning. The graph-based planning models (AoA and AoN networks) often used for project scheduling are no longer able to present flexible relationships between project tasks.

Flexibility is championed by matrix project planning methodologies, such as the widely-used Multi-Domain-Matrix (MDM) or Design Structure Matrix (DSM) (Browning 2015). This approach has been applied to present flexible dependencies between project activities in case of various flexible projects (Kosztyán and Szalkai 2020, Kosztyán et al. 2022, Kosztyán 2022). Both the management and methodological aspects can be effectively addressed by utilizing domains in a multidomain mapping matrix (Browning 2014, 2015) to plan and coordinate all the requirements, synergies, and task-employee dedications.

However, the CPM and PERT method became obsolete not only because of the appearance of flexibility. In developing countries, the successful implementation of projects serves as a cornerstone for national development and plays a pivotal role in driving economic growth and improvement (Habibi et al. 2018). But, during the conceptual change of project management (see Section 2.1.3) and the evolution of project success (see Section 2.1.5), the management had to face more and more constraints during the scheduling of projects with the increase of globalization and global competition.

The resource-constrained project scheduling started when efforts were made to address the limitations posed by resource constraints in mathematical problem formulations by Pritsker et al. (1969). However, significant attention was only garnered in the last quarter of the 20th century. These emerging challenges were coined as Resource Constrained Project Scheduling Problems (RCPSP), representing a delicate balance between time and cost constraints and provides a formalized framework for addressing both task precedence and the allocation of limited resources (Hartmann and Briskorn 2022). While traditional approaches focused solely on minimizing project time by identifying the critical path through the shortest spanning tree in a project-as-graph model (where activities are defined by network structure), RCPSP broadens its scope to accommodate diverse resource constraints, such as capital or human resources, which may be renewable or non-renewable (Li and Zhang 2013, Habibi et al. 2018).

2.2.1 Resource constrained project scheduling problems

RCPSP is described as follows (Blazewicz et al. 1983, Hartmann and Briskorn 2022): There is a project consists of J activities, labeled $j = 1, ..., J$, that can be characterized by a sequence of dependencies. The duration of an activity is denoted by p_j, and an activity must not be interrupted if it has been started. J can only be started if its predecessors - labeled P_j - are completed. The first activity is the START and the last is the END with 0 needs. No activity can be started before the START and finished after the END. K resources ($k = 1, ...,$

K) are considered with a constant per-period availability R_k. Each *J* activity requires $R_{j,k}$ units of resource *k* during its execution. All parameters are deterministic, non-negative, and integer-valued.

A schedule assigns a non-negative start time S_j to each activity *J = 0, 1, ..., J+1*. The goal of the RCPSP is to determine a schedule that minimizes the project's makespan, defined as the completion time of activity *J+1*.

The outcome of RCPSP is a collection of potential optimal scheduled work breakdown structures by minimizes the project's makespan, defined as the completion time of activity. All while adhering to specified resource constraints. This complexity positions RCPSP as an illustrative example of NP-hard problems, as demonstrated by Blazewicz et al. (1983). RCPSP has been extensively explored (Hartmann and Briskorn 2022), along with its solution algorithms, as highlighted by Habibi et al. (2018). Problems such as the interruption (Vanhoucke and Coelho 2019) or optional execution (Kellenbrink and Helber 2015) of certain tasks, the need to add additional resources (Zimmermann and Trautmann 2018) or changes in resource capacity (Kreter et al. 2016) and slippages such as setup time (Hanzalek and Sucha 2017) or time lag (de Azevedo et al. 2021) have been modeled during scheduling. Furthermore, the objective function was extended to trade-off problems, such as the trade-off between time and cost (Leyman et al. 2019), and robust design (Hazır et al. 2015) was also taken into account. In addition, more and more complex RCPSP papers have been published lately and the basic RCPSP model remains one of the most scrutinized areas in the literature. On the other hand, the RCPSP also has limitations, which can no longer deal with the problems of today's world. Therefore, the literature now builds on various extensions of the RCPSP, since agile methodologies are now indisputably embedded in the life of project management, emphasis must also be placed on the human side of project scheduling (Meckenstock 2024). In today's world, high turnover mainly affects projects where their schedule depends significantly on the capabilities of the project team.

These complex extensions to the RCPSP induced the appearance of new basic models. The capabilities of team members were historically overlooked as a restriction, until the recent emergence of MS-RCPSP (Myszkowski et al. 2017). First appearance of MS-RCPSP is in article by Hegazy et al. (2000) to solve the replacement of employees. The new method was proposed based on the CPM network and compared with single-skilled RCPSP, during which a better total project time was achieved with MS-RCPSP. In the context of MS-RCPSP, activities necessitate specific skills for execution, and the human resources have these multiple skills. Importantly, the introduction of skills does not alter the precedence relations between activities (Snauwaert and Vanhoucke (2023)). In the past few years, numerous extensions of the MS-RCPSP were implemented, which have been comprehensively outlined in the works of (Snauwaert and Vanhoucke 2023, Hartmann and Briskorn 2022). Their study delves into various adaptations and enhancements of MS-RCPSP, shedding light on the evolving landscape of this complex scheduling problem. In terms of capability improvements, hierarchical levels of skill (*l = 0, ...,L*) was proposed by Bellenguez and Néron (2004), Myszkowski et al. (2015a), Zhu et al. (2021), which can be used to model not only the type of capabilities, but also the distribution of individual capabilities in the team. In addition, learning and forgetting effect was presented by Chen et al. (2020). Also personalities and soft skills are modeled by Akbar et al. (2022), using the MBTI personality model and separating the hard and soft skills. In addition to the importance of skills, development was also made in the direction of project portfolio level by (Kolisch and Heimerl 2012, Chen et al. 2020)

Another model is the Multimode RCPSP (MRCPSP) (Coelho and Vanhoucke 2011) individual tasks can be performed in different ways or by using different resources (renewable or non-renewable). Multi-Skilled Multimode project scheduling problem (MS-MRCPSP) with makespan was introduced by Maghsoudlou et al. (2016), as a merger of MRCPSP and MS-RCPSP. This method has been further developed to handle the changes in professional capabilities caused by digital transformation (Li et al. 2024a). Although the same variables and parameters are defined as in the case of RCPSP, additional ones are included here, such as the index of execution modes (*m = 1, ...,M*) and index of skills ((*k = 1, ...,K*)), while resources are denoted by (*s = 1, ...,S*). Therefore, in the case of the MS-MRCPSP, the tasks, the dependency relationships between them, the resources and capabilities needed to solve the tasks in each mode must be defined as inputs.

Since MS-RCPSP and its extensions are based on RCPSP the problem is NP-hard, therefore several heuristic algorithm were developed to solve this, like branch-and-bound algorithm (Bellenguez-Morineau and Néron 2007), tabu search algorithm (Drezet and Billaut 2008), invasive weeds algorithm, a non-dominated sorting genetic algorithm or particle swarm algorithm (Maghsoudlou et al. 2016).

To evaluate and assess each new variant of the MS-RCPSP, researchers can turn to the extended iMOPSE database introduced by Myszkowski et al. (2019). This database serves as a valuable resource, providing a platform for checking and testing the performance of different MS-RCPSP types. The iMOPSE database contributes to the systematic analysis and benchmarking of various MS-RCPSP extensions, offering researchers a standardized and reliable means of comparison. It is even good for testing the correctness of algorithms with it, as it can be seen in Myszkowski et al. (2015b).

RCPSP and its extensions are well able to model all those projects where at least a lower and upper value can be linked to the completion time of each activity. Specifically in the software industry, the MS-RCPSP is more complex due to the strong dependence of the performance of the tasks on the characteristics of the human resource. Since in industry, especially in the field of software development, the time of individual activities depends on the skills of the software developer working on it, it is difficult to give an estimate of the time of software development activities. Therefore, for the efficient scheduling of such - especially software - projects, a method was needed that takes into account the activity times not predefined but as a function of human abilities and optimizes the trade-off problem between time and the cost of people' salary.

2.2.2 Software project scheduling problem

In the literature of software projects the Software Project Scheduling Problem (SPSP) was formed (Alba and Chicano 2007). It becomes evident that SPSP closely aligns with MS-RCPSP, sharing similar limitations and deficiencies. In both SPSP and MS-RCPSP, task duration is contingent upon the skills of employees, as specified in the basic framework of SPSP by Alba and Chicano (2007). They define fixed or predefined precedences between tasks, as elucidated by Luna et al. (2014). While skills can vary based on the resource requirements of tasks, they are often not predefined due to the limited implications of such methods. The key distinction lies in the fact that while MS-RCPSP solely considers resource limitations, including the required skills for each task, SPSP goes further by determining the assignment of human resources to tasks. Therefore, in the case of SPSP, the duration of tasks is usually not predetermined, but depends on the abilities of the people working on the task. Furthermore, as it was written by Alba and Chicano (2007) another main difference is that MS-RCPSP and other RCPSPs generally only optimize for duration minimization, while SPSP also deals with cost minimization at the same time (and sometimes maximizing score (Kosztyán et al. 2022) or quality (Hanne and Nickel 2005)), which depends on the salary of each employees.

The basic framework of SPSP is still applicable today, as depicted in their general model (Figure 8). According to this model, the scheduling of software projects hinges on the characteristics of employees and activities. Optimal employee assignments to activities are based on their skills, the resource requirements of the activities, the maximum dedication of employees to each activity, the effort needed for each activity, the precedence between activities, and employees' salaries. The solution to SPSP facilitates the determination of employee assignments to activities and the presentation of scheduling through a Gantt chart.

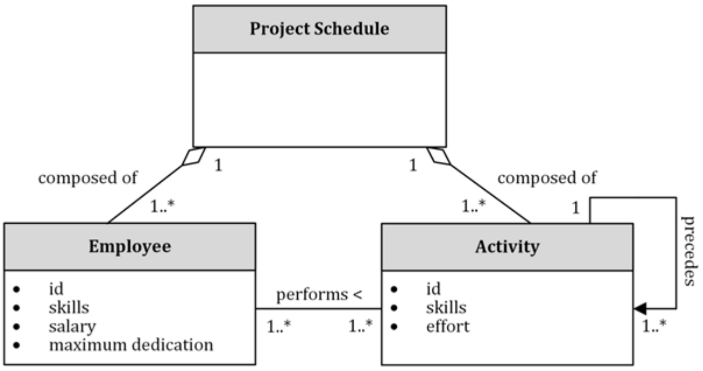


Figure 8: Basic scheme of the SPSP - UML diagram

To have the optimal solution the following constraints must be complied:

- C1. Each activity must be performed at least one employee.
- C2. The required skill set from an activity must be a subset of the union of assigned employees' skills.
- C3. Maximum dedication of each employee must not be exceeded.

As the MS-RCPSP, the SPSP has been also extended in several ways. Since the two models are very similar, in practice the same improvements can be made to both. While the fundamental model is quite adept at representing SPSP, not all studies have comprehensively addressed every aspect of it, as evidenced by differences highlighted in Vega-Velázquez et al. (2018).

Diverse extensions have been explored concerning objectives and applied optimization methods, as investigated by Vega-Velázquez et al. (2018) and further complemented by Rezende et al. (2019). Although the two systematic literature searches summarize the literature and methods published in the field of SPSP up to 2018, it is also important to present the studies after 2018 to be able to understand the model developments resulting from the current economic situation. For this, I conducted a systematic literature review using the PRISMA method (Moher et al. 2009), which basically consists of 4 parts: (1) identifcation, (2) screening, (3) eligibility and (4) inclusion. Detailed method is presented in Appendix A.

Based on the literature review (Appendix A: Table 39), a similar process of expansion of the SPSP is seen as in the case of the RCPSP. The SPSP followed the evolution of the interpretation of the project and the changes in the economy. Initially around the 2010s, the main focus was the development of the time-cost trade-off (Alba and Chicano 2007, Chicano et al. 2012, Suri and Jajoria 2013), and later on the development of the most efficient algorithms. Like MS-RCPSP, SPSP is also an NP-hard problem (Xiao et al. 2013a), so that there are also various metaheuristic algorithms which have been employed for its solution, analogous to the approach taken for MS-RCPSP. In general, these NP-hard SPSPs are solved using algorithms that mimic behaviors seen in nature. Among the solution algorithms, the non-dominated sorting Genetic Algorithm (NSGA-II) (Deb et al. 2002) from the evolutionary algorithms family stands out as the most popular. But, there are several other algorithm for solving the problem like the intelligent water drops (Crawford et al. 2018) or the grey wolf optimization (Alabajee et al. 2021). However, other algorithms have been explored for solving SPSP, as documented in studies by Vega-Velázquez et al. (2018) and Rezende et al. (2019).

Then, after the 2010s, uncertainty in software development was increasingly realized (Mehta et al. 2014), so methods were needed that can handle this during scheduling. These generally support a schedule where unexpected events are expected. Such as Szwarc et al. (2024), who models unexpected sick leave or a change in the priority of tasks. According to another approach (Li et al. 2024b), the time of the task itself may be uncertain, or it may also happen that the resource changes dynamically (Shen et al. 2015). Due to the indeterminacy caused by the uncertainty, a more dynamic schedule had to be developed than for RCPSP or MS-RCPSP. Some called it robustness (Gueorguiev et al. 2009), others interpreted it as stability (Xiao et al. 2010) but often the two concepts manifest in one model (Cheng et al. 2019, Nigar 2017, Shen et al. 2015, 2018, 2020, Silva et al. 2020). Moreover, some other aspects like flexibility and agile was considered by Zapotecas-Martínez et al. (2020). However, dynamic scheduling has increasingly focused on individual-focused dynamics, mainly after the introduction of agility. The effect of multi-skills (Li et al. 2023, Kosztyán et al. 2022) and level of skills (Kosztyán et al. 2022, García-Nájera and del Carmen Gómez-Fuentes 2014, Duggan et al. 2004) on scheduling were investigated, as could be seen in the case of the MS-RCPSP. They also modeled the dynamics of capabilities. In these models (Cheng et al. 2019, Guo et al. 2019, Nigar et al. 2022, Szwarc et al. 2023, 2024, Zhang et al. 2023), existing skills change depending on how long a given employee uses them, so they can learn and forget them. But there were also examples (Shen et al. 2024) where employees could learn certain skills during the time they spent with other employees.

In addition to abilities, different human behaviors such as behavioral types (Stylianou et al. 2012, Stylianou and Andreou 2013) were also taken into account, although in this case purely in terms of abilities and missed to use any kind of behavioral types theory. The effectiveness of communication was also modeled (Ge and Bin 2016, Zhang et al. 2023), and steps were taken to model the soft skills and the conception of team as a graph. With the introduction of agility, the importance of teams became more and more important, as they realized that with the increase in complexity (Persson and Mathiassen 2009), employees are no longer able to perform individual tasks efficiently alone. Thus, phenomena such as team synergy (Kosztyán et al. 2022), which represents the effectiveness of joint work, were also modeled, where the authors also mention the difficulty of defining synergies. They proposed the synergy-based software project scheduling problems (SSPSP), integrating the matrix-based flexible software project planning with pairwise synergies between employees.

Software project scheduling problems have been widely studied, during which several, often unrelated, extensions of the general model have been made.

2.3 Summary of the evolution of project management

Over the past 30 years, the interpretation of project management has evolved significantly. In its early stages, projects were universally defined by the iron triangle, a framework focused on balancing time, cost, and scope to achieve a known goal. During this period, traditional project management methodologies dominated, prioritizing efficiency and resource optimization. Techniques such as the Resource-Constrained Project Scheduling Problem (RCPSP) and its extensions emerged to address the challenges of resource scarcity, emphasizing outcomes over process adaptability.

With increasing globalization reshaping competitive landscapes, companies began embedding projects into their strategic frameworks and viewed as temporary organizations, and their success criteria expanded to include stakeholder satisfaction and alignment with project owner's goals. This shift led to the rise of more flexible, iterative approaches, as businesses recognized the need for adaptability in an increasingly dynamic environment. Despite the fact that previously it was necessary to quickly adapt to changing customer requirements by involving the customer more closely, the transition to agile methodologies was slow and often driven by necessity rather than proactive strategy. Consequently, traditional approaches, including enhanced scheduling models like the Multi-Mode Resource-Constrained Project Scheduling Problem (MRCPSP), remained dominant in the context of projects as temporary organizations.

The appearance of digitization and the rapid acceleration of economic development brought new challenges and opportunities. The software industry, in particular, became the base for agile project management approach, as organizations needed to adapt quickly to changing market demands and replace legacy products with more cost-effective solutions. Agile methodologies gained traction by emphasizing flexibility and customer focus, aligning well with the industry's fast-paced environment. Scheduling methods like the Software Project Scheduling Problem (SPSP) emerged, reflecting the dual priorities of time and cost in a flexible environment.

However, the adoption of agile methodologies has not been without its challenges. Studies, such as those by Meckenstock (2024) or VersionOne (2024), have highlighted pitfalls that can undermine agility in organizations, including turbulent economic conditions, resistance to change, and inadequate team structures. These challenges underscore the limitations of purely agile approaches, prompting the rise of hybrid methodologies that blend traditional and agile principles to better address organizational complexities. Despite these advancements, many project scheduling models continue to prioritize traditional success criteria, with limited consideration for the evolving demands of agile environments.

The COVID-19 pandemic marked a turning point in corporate strategy. Companies began to prioritize resilience, focusing on warehousing, workforce retention, and flexibility to navigate supply chain disruptions and economic uncertainty. This shift was particularly significant in software development, where human capital is the primary resource. As organizations recognized the critical importance of retaining experienced employees and employ self-organization during team selection, the focus shifted from individual capabilities to team dynamics and open a new way of researches. While agile methodologies are inherently well-suited to software development, the applied scheduling methods often overlook the nuanced interactions and hidden mechanisms within teams.

This gap has been partially addressed by the Synergy-Based Software Project Scheduling Problem (SSPSP), which incorporates pairwise synergy effects between team members into project scheduling. However, as Kosztyán et al. (2022) pointed out, the estimation of these synergy networks remains an open question. To model team dynamics effectively, it is essential to integrate behavioral types theories and understand the behaviors of diverse team members, since both determine the success of the common work (Xu and Correia 2024). By combining these insights with the foundational principles of SPSP, it becomes possible to extend the SSPSP framework and better account for team-level interactions. The new project scheduling method is expected to help understand the dynamic capabilities caused by changes in team roles and the efficiency of autonomous groups. With the former, we can provide a more robust project schedule for software development struggling with a lack of resources, and the latter helps to shed light on one of the important foundations of agility, the self-organization.

In conclusion, the evolution of project management (see in Table 5 reflects a wider shift from rigid frameworks to adaptable, team-centered approaches. While early methodologies prioritized resource optimization and clear deliverables, today's dynamic environment demands a deeper understanding of team dynamics and human behavior. For software projects, where people are the most valuable resource, incorporating these factors into scheduling models is essential. By studying and integrating team dynamics, the next generation of project scheduling methodologies can align more closely with the realities of modern software environments, ensuring both flexibility and resilience.

Table 5: Summary table on the evolution of the interpretation of project management

Time	Antecedent	Projects concept	Project cycle	Success criteria	Critical success factors	Project Management Approaches	Project scheduling method
- 1990	Resource constrains due to the high volumes after the 2 World War	Bounded by time, cost, quality	Traditional Waterfall	Iron triangle	Project mission, scheduling, politics, stakeholders, priority, project manager	Traditional PM	RCPSP
1990-2005	maximizing the value created by the projects became the main goal due to the resource constraints and increasing competition caused by globalization	Strategic oriented projects	Iterative V model	Strategy	12 success factors by (Cooke-Davies 2002)	Traditional PM	MRCPSP
2000-(2021)	rapid changes in customer needs and increasing competition due to the digitization and innovation required flexible solutions	Project as a temporary organization	Evolutionary model	Stakeholders	Involvement, support, communication, knowledge and technical expertise	Agile PM	MS-RCPSP
2012-(2021)	adapting rapid digital solutions with data-driven decision making	Agile portfolio projects	Hybrid model	Agile	People, knowledge, sustainability	Hybrid process-, customer-oriented PM	SPSP
2021-	the recognition of economic risks, the shortening of supply chains as a result of the economic crisis caused by the Covid-19 pandemic	Hybrid sustainable-, and people focused projects	Hybrid model	Process efficiency, sustainable software product quality, stakeholder satisfaction	Project owner, team, organization, agile process, technical, project	Hybrid people-oriented PM	Extended SPSP with team efficiency

2.4 Team Selection

The characteristics of the project members are not only inputs to the scheduling of software projects, but also fundamentally determine the success of them. However, it is not enough to schedule based only on the characteristics of individuals, since software projects are basically carried out by groups/teams. Learning about effective teamwork is a long-researched field, and to this day many articles are published, mainly in the fields of psychology, sociology and occupational psychology. Research on the role of teamwork and collaboration can be attributed to famous people such as Bruce Tuckman, Richard Hackmann, Kurt Lewin and Meredith Belbin. Before I get to their most significant results, it is worth separating the concepts of group and team from each other.

The Cambridge dictionary defines a **group** as *a number of people or things that are put together or considered as a unit* , while Merriam-Webster dictionary defines it as *A number of individuals assembled together or having some unifying relationship.* The most relevant definition of a team according to this thesis is a little different by the Cambridge dictionary, that the **team** is *a number of phrases that refer to people working together as a group in order to achieve something*

An individual's behavior and performance can differ significantly when working in a group versus in a team, making it both interesting and essential to study team behavior and dynamics. An example of this could be if the team only consists of experts, but the communication channels between them are weak, so their ability to cooperate is also low. But on the other hand, even a team of average performers can bring about extraordinary results with good synergy (Groysberg et al. 2011).

The composition of a team can be studied from varying perspectives (Guimera et al. 2005, Bell and Outland 2017, Bell et al. 2018a). If the software development team is considered, it can be characterized by only basic demographic characteristics (age, sex, experience etc.), but it can also be described by deeper characteristics such as synergy (Hertel 2011), behavioral types (Soomro et al. 2016), skills and abilities (Saldaña-Ramos et al. 2014), attitude (Matthies et al. 2019) or team cohesion (Grossman et al. 2022).

But overall, it can be said that a good team usually needs some time before start working efficiently, since the group of people is not a team. An effective division of responsibilities is the key to a team's success. When each member assumes a role that aligns with their skills and personality traits, it serves as a strong indicator of a well-functioning team. So that the basic characteristics of individuals will affect and determine the characteristics of the group and then the team. However, for this they must first form a group, which can be selected based on certain criteria.

The process of selecting software development groups is a critical aspect of successful project management in the dynamic landscape of software development. The decision-making in this context involves careful consideration of various factors to ensure the effectiveness, efficiency, and overall success of the development process. Effective software development groups are characterized by strong interpersonal relationships, clear communication channels, and a collaborative culture. During a selection mechanism, the members of a group are selected, who will form a team during the formation of the group. However different tasks may require different group structures and processes. The expertise and abilities of group members are pivotal to success. A well-balanced combination of skills and knowledge is essential for optimal performance. However, possessing high levels of expertise alone is insufficient; groups must also be empowered to apply that expertise effectively. Without proper structure or support, groups can become passive and under-perform. Moreover, the physical setup and broader social environment play a significant role, either fostering or impeding collaboration (Hackman and Katz 2010).

The phenomenon that examines how groups made up of individuals become teams and how individual persons find their place in this organized unit is examined by the so-called occupational psychology (in other words: work psychology or industrial-organizational (I-O) psychology) (Steptoe-Warren (2013), Carruthers (2000)). From the theory of group development (Tuckman (1965)), which is one of the greatest example of the occupational psychology, the four well-known stages of it: forming, storming, norming and performing. In the forming stage, only individuals are considered, i.e. a company of people with different personality traits who will presumably cooperate as a team later on based on some structure. During the forming phase, the success of later phases by optimally selecting the team can be facilitated. The difficulty is caused by the storming phase, where the process of forming a team begins. In this phase, all contradictions caused by bad selection or bad management decisions come out. This also includes the contradictions arising from differences, which naturally follow from different upbringings and behavioral types. By the time when the norming stage is got, most of the conflicts have already been resolved and the group starts working on the tasks as a team. The transformation of individuals can also be observed: while their individual interests have been at the center of their preferences and therefore their decisions until now, collective thinking, group decision-making and the representation of the group's interests begin to develop in the norming phase. Every stage has its importance, however, when scheduling software projects, the emphasis is on the formation of the team because the scheduling is highly rely on the corresponding group selection. This is presumably finalized in the norming phase, where team roles are normalized and the team begins to function as an organized unit. Of course, in practice, the scheduling of software projects begins already in the forming phase of team development. for this reason, it is important that the schedule implemented in this way focuses on features that are as robust as possible, which are either features innate to the individual, or features that require a longer time to change.

In order to gain a deeper understanding of the factors affecting occupational psychology, it is necessary to understand the characteristics of individuals and the differences between them. The further subsections of the dissertation focus on these.

2.4.1 Individual diversity

According to the definition of **individual** by the Cambridge dictionary, it is a *single person or thing, especially when compared to the group or set to which they belong.*

The definition of the group shows that its smallest unit is the individual, with all its properties. So, the properties of a group are determined by the properties of the individuals in the group, while the properties of a team are also determined by the relationships between the individuals. According to Lewin (1947), each team member brings unique strengths and expertise to the group, reflecting the diversity of individuals. These varied abilities are complementary, and when effectively harnessed, they enhance the team's overall capacity to tackle a wider range of tasks and challenges more efficiently. A diverse team, made up of different behavioral types and cognitive styles, is better equipped to analyze problems from multiple perspectives, leading to more thorough discussions and more effective decision-making. Although differing viewpoints can sometimes result in disagreements, they also create valuable opportunities for constructive conflict. When managed properly, these conflicts can deepen understanding and ultimately strengthen team cohesion.

Jehn et al. (1999) classify team diversity into three primary categories: social, informational, and value diversity. Social diversity encompasses explicit differences among team members, such as ethnicity, gender, and age. Informational diversity refers to variations in knowledge and expertise, while value diversity highlights differences in beliefs regarding the team's mission, goals, and objectives.

Diversity can shows positive effects on performance of teamwork (Peslak 2006, Bear and Woolley 2011, Galinsky et al. 2015), because of compatibility of work habits (Kang et al. 2006), mindful communication (Phillips et al. 2009), higher motivation (Katzenbach and Smith 2015) or higher problem-solving competency (Higgs et al. 2005). But some authors have the opposite view (Towry 2003, Van Knippenberg et al. 2004), because of additional costs (Hamilton et al. 2012), clicking due to the differences and similarities among team members (Waleed et al. 2021) or communication problems and conflict Fincher et al. (2001), Ojha (2005).

Perhaps it is not surprising that there are both positive and negative opinions about team diversity. Upon reflection, the effectiveness of team diversity depends on whether complementary people succeed in forming a team. From the above, it can be concluded that the success of a diverse team is more likely to be achieved if the team members are highly motivated, have complementary skills and have good team synergy, which is basically influenced by the existence of behavioral types.

2.4.2 Individual motivation

People, as beings, need certain basic needs without which they cannot be motivated. This is well modeled by Maslow (1943) pyramid model, which, despite receiving a lot of criticism, demonstrates well that a hierarchy can also be established among human needs. At the bottom of the pyramid are basic needs such as physiological needs or the desire for safety. But beyond that people want to get more incentives that makes them motivated and committed, which is much more individual. For example, if a person's main motivation is social relationships, then even with all the basic needs and a high salary, she/he will be demotivated if she/he does not have suitable companions at work. The degree to which team members are attracted to each other and motivated to stay in the team can significantly impact performance (Lewin (1947). This conception is represented by the Motivation-Hygiene Theory model (Herzberg (1966), Herzberg (2008)). Motivation has been considered as a key factor for the satisfaction of the software engineering (França et al. (2011)). Therefore, a lack of motivation can cause the inactivity of them. For the Software Engineers these motivators can be different than for the other people. Beecham et al. (2008) found that based on their characteristics the main motivators of the software engineers are the continuous advancement (Growth Orientated peoples), the autonomy (Autonomous team) and the separated workspace (Introverted peoples) and the most important control factor is their personality traits. However, they want to identify with the tasks, work with other employees, have a good management and avoid the retention. This finding has been partially confirmed by França et al. (2011) but focused more on the evolving agile environment. Turning to the agile methods the motivators have been studied more. Melo et al. (2012) found that technically challenging work and teamwork are the key motivators. But I need to mention that the development needs addressed presented by all examined companies, but it was not frequent.

2.4.3 Skills and abilities

Skills of the project team members can be separated into hard and soft skills (Balcar 2016, Napier et al. 2009).

According to the Cambridge dictionary:

- *soft skills:* an ability that does not depend on knowledge needed for one particular job, but on, for example, being able to work well in a team, communicate well with people, etc.
- *hard skills:* Are competencies that employees possess such as numeracy, literacy, fluency in a foreign language, and specific job-related technical abilities (operating a machine, creating a spreadsheet, touch-typing, driving, dressing a wound, and so forth). Typically these skills are relatively easy to measure, and are often validated with some form of qualification. More recently, there has been a shift in emphasis towards the need for soft skills in addition to technical abilities.

Based on Pant and Baroudi (2008) 67% of software projects failed because of soft skill issues. In software development environment the only resource is the human capital except of the fixed assets. I assume in the SPSP (based on SPSPs) that the performance of the human resource depends on their skills. Citing Sánchez-Gordón et al. (2020) in recent years, it is increasingly discussed in the software engineering community that technical, also known as hard skills, and non-technical, also known as soft skills, are equally important as software is developed by people for people (Garousi et al. 2019). The job performance is depending on both technical and soft skills which is confirmed by Mtsweni et al. (2016) (Figure 9. Project team members need to have technical knowledge to work on the project tasks however they should be working together as a team which requires high level soft skills.

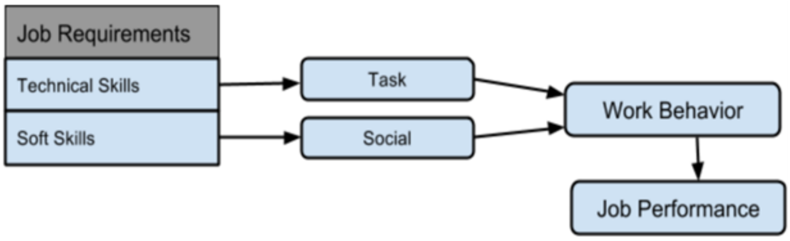


Figure 9: The effect of the job requirements on job performances. Source: Mtsweni et al. (2016)

According to Sukhoo et al. (2005) the hard skills (like programming) are learnable by using processes/tools or techniques. In contrast the soft skills are not or hard to learnable skills that have very strong relation to the evolution of a person. Despite of the soft skills cannot be learned; team members are able to develop themselves to increase their hard skills. Every person has an own learning and forgetting skill. During the software project scheduling learning and forgetting skills can have impact on the scheduling as some papers mentioned (eg.: Guo et al. (2019), Cheng et al. (2019), Nigar et al. (2022)). Therefore, the researchers improved these models to be able to describe the continuous learning and forgetting based on the assignment of the employees to the project tasks. I must emphasize that only the hard skills can be considered into the learning and forgetting models. In regards of the software development team which is normally contains a project manager, requirement analysts, software developers and software testers a small skill set has the biggest impact on the success of the project although there are differences between the importance of these skills for each role. This skill set can be very specific in different projects; however, Hidayati et al. (2020) and Matturro (2013) suggest that the required hard skills are the Programming Skills; Scrum Expertise; Specific Skills; Spoken and Written Language Skills and Database knowledge.

However hard skill performances usually depend on former experience, therefore it is more project specific than the soft skills. I can consider these skills as the base of the hard skill set to complete a software development project. Besides that, I can find much more paper about the required soft skills for the software development team. On the one hand the required soft skills are not depending on the type of the software but on the other hand the lack of the soft skills can have higher negative input on the success of the software project than the lack of the hard skills. That's why I must consider soft skills as stricter than the hard skills. To define the soft skill, I can use the composition by Matturro et al. (2019).

- Abilities: Competence in an activity or occupation because of one's skill, training, or other qualification.
- Attitude: A predisposition or tendency to respond positively or negatively towards a certain idea, object, person, or situation.
- Habits: An acquired (learned rather than innate) behavior pattern regularly followed until it has become almost involuntary.
- Personality traits: An acquired (learned rather than innate) behavior pattern regularly followed until it has become almost involuntary.

Based on their analysis of the frequency of the soft skills mentioned in papers about software teams, the first 5 most common soft skills are communication skills, teamwork, analytical skills, organizational/planning skills and interpersonal skills. On this basis, I can conclude that these are the most important soft skills for the software project team members.

Before that survey Mtsweni et al. (2016) identified the key soft skills which are the team player, the personal integrity, the group work, the time management and the effective questioning, although they also investigated these key soft skills separately for the most important roles of the software projects. Later Omar et al. (2018) listed all soft skills that are required for the software team in agile environment. These are the analytical skills, communication, facilitation skills, interpersonal skills, leadership skills, management skills, people skills, planning skills, teamwork skills and thinking skills. Furthermore, based on Hidayati et al. (2020), Matturro (2013), Borges and de Souza (2024) the top five soft skills are the interpersonal and communication, teamwork, analytical thinking, management and planning and leadership. I can recognize that the number of soft skills required by the software project can be different however I can say that the following soft skills can be generally considered as mandatory skills for the software projects:

- Communication ability/skills
- Leadership ability/skills
- Teamwork attitude
- Problem-solving skills
- Analytical thinking
- Interpersonal skills

Further important skills are the “commitment, responsibility”, “eagerness to learn”, but they are more individual specifics than the others. Team member may possess unique skills that complement one another. For example, some may excel in analytical thinking, while others may be strong in creative ideation or interpersonal communication. This combination of skills can enhance the team’s overall effectiveness (Hackman and Katz 2010). Although this is not sufficient, the team must also be able to exploit and coordinate its inherent abilities (Faraj and Sproull 2000).

2.4.4 Behavioral types and team roles

Diverse teams tend to excel at addressing complex challenges, as they can tap into a broader spectrum of knowledge and perspectives. This diversity fosters deeper analysis and more effective solutions. A blend of different personal types also enriches team interactions and discussions, strengthening both cohesion and morale. When individuals feel appreciated for their unique contributions, it creates a more positive and supportive team atmosphere (Hackman and Katz 2010).

According to the Cambridge dictionary the meaning of personality is defined as *the type of person you are, shown by the way you behave, feel, and think.* Another definition is given by the Oxford dictionary: *The combination of characteristics or qualities that form an individual’s distinctive character.*

The literature lists many theories of personality types or traits (Boyle et al. (2008). In the field of software engineering, the field of pair programming is where the effects of different behavioral types have been investigated. Another important area is the effect of behavioral types on team effectiveness, so that how it can be affected by personality interactions among team members (Cruz et al. 2015). Without the full picture, the most well-known personality and behavior theories are the Myers-Briggs personality library Myers (1962), DISC behavioral types (Marston (1928)), Big-Five model (Costa and McCrae (1999)), Enneagram of Personality (Riso and Hudson (2003)) or Jungian Archetypes (Weiner et al. (2019)), while the most well-known team role theory is the Belbin’s team role theory (Belbin (1981)). Personality traits and behavioral types (and soft skills) has a massive effect on the productivity of the organization (Norhanim et al. (2019)) and can contribute to the success of the project team (De Vreede et al. (2012)).

Among them, the theory of DISC behavioral types is more flexible due to the small number of different personalities (only 4). Because of its simplicity and popularity, the DISC behavioral types have gradually increased in recent years and can be used for the selection of small agile teams (Diekmann and König 2018, Reynierse et al. 2000, Lykourentzou et al. 2016)). DISC was founded in 1928 by William Moulton Marston, who separated behavioral types along two dimensions (task/people orientation; extroversion/introversion) and measured four aspects of human behavior (see Table 6). While the allocation of the different DISC behavioral types can highly influence team performance and well-being (Lykourentzou et al.

2016), Antoniou (2019) found that there is no relation between balanced and unbalanced teams.

Table 6: Brief description of DISC behavioral types

Behavioral type	Characterization	Main properties
Dominance	extrovert, task-oriented, the leader of the team, takes responsibility	active, powerful, confronting, obstinate
Influence	extrovert, people-oriented, the soul of the team, maintains motivation	friendly, optimistic, easy-going, unstable
Steadiness	introvert, people-oriented, the parent of the team, maintains stability	team-player, reliable, loyal, retractive
Conscientiousness	introvert, task-oriented, the brains of the team, precisely promotes solutions	accurate, analytical, perfectionist, mistrustful

With DISC methodology a better shape of the project teams is given, however DISC profiles are related only individual members. Behavior of team members in a team can be described better by the Belbin's team roles. Team role is defined as a cluster of behavioral characteristics which individuals display when working in teams (Belbin 2012). The organization of roles and responsibilities within the group can affect how effectively it operates. Clear roles can enhance performance by providing direction and reducing confusion (Lewin 1947). It is important to note, that there is no correlation between behavioral types and team roles (Diab-Bahman 2021). Meredith Belbin publicized his research in 1981 firstly with Management Teams title. Since then, several revised editions of the book have been published (e.g.: Belbin (2012)). Belbin describe that different team roles increase the possibilities of the team and reduce the risk of conflict between persons who can fulfill the same function. However, it is hard to break up a stable team because the team does not tolerance any new people. He mentions that during his analysis there was not any relationship between the work morale and the result of the teams. He says that there was a team that went bankrupt happily and with good cooperation. This suggests that although the relationship between team members is good, the effectiveness of joint work is influenced by other factors (see in Section 2.4.5). He also comments that almost 30% of the participants cannot work within a team. Based on his analysis 8 (and later 9) different team roles can be separated (see in Table 7).

Table 7: Definitions of Belbin's team roles

Type	Name	Belbin (1981)	Fisher et al. (2001)
CO	Coordinator / Chairman	Mature, confident, a good chairperson; clarifies goals, promotes decision making; delegates well; inclined to be lazy; takes credit for effort of a team.	Delegates tasks and rallies team; makes firm decisions usually having consulted others; adaptive to changes; committed to goals and sees to them being met; uses all team members making all play a role; draws out the potential from all members; recognizes talent and uses it
TW	Teamworker	Co-operative, mild, perceptive, and diplomatic; listens, builds, averts friction, calms the waters; indecision on crucial issues; avoiding situations that may entail pressure.	Listens to others and supports them; works with the awkward people; not forceful or demanding; diplomatic and balancing; averts conflict; communicates well with others.
RI	Resource Investigator	Extrovert, enthusiastic, communicative; explores opportunities; develops contacts; loses enthusiasm once initial excitement has passed	Negotiates and liaises with outsiders; asks questions from others; opportunistic; thinks on feet; picks up on the ideas of others; is sociable; inquisitive and curious; enthusiastic about tasks at the beginning.
IMP	Implementer / Company Worker	Disciplined, reliable, conservative, and efficient; turns ideas into practical action; adherence to the orthodox and proven; obstructing change.	Orderly and precise; sticks to rules; doesn't like change; organizes plans and action; disciplined in approach; systematic in approach; tackles any tasks.
CF	Completer-Finisher	Painstaking, conscientious, anxious; searches out errors and omissions; delivers on time; perfectionism; obsessional behaviour.	Thorough about a task; attentive to all details; finishes things; reluctant to let go until complete; nags others to finish on time; perfectionist; plans so that nothing gets overlooked.
SH	Shaper	Challenging, dynamic, thrives on pressure; has the drive and courage to overcome obstacles; a proneness to frustration and irritation; inability to recover situation with good humour or apology	Thinks things through; tackles problems usually on own; proposes ideas and solutions; creative and imaginative; dominant with ideas.
PL	Plant	Creative, imaginative, unorthodox; solves difficult problems; preoccupied with ideas and neglects practical matters; strong ownership of ideas	Thinks things through; tackles problems usually on own; proposes ideas and solutions; creative and imaginative; dominant with ideas.
ME	Monitor Evaluator	Sober, strategic, discerning; sees all options; judges accurately; scepticism with logic, cynicism without logic.	Asks for all information; unenthusiastic and impartial about ideas; slow to make a decision; likes to think based on all facts; negative about plans

Belbin categorized the team roles and form 3 different group: thinking-oriented roles, action-oriented roles and people-oriented roles (Figure 10).

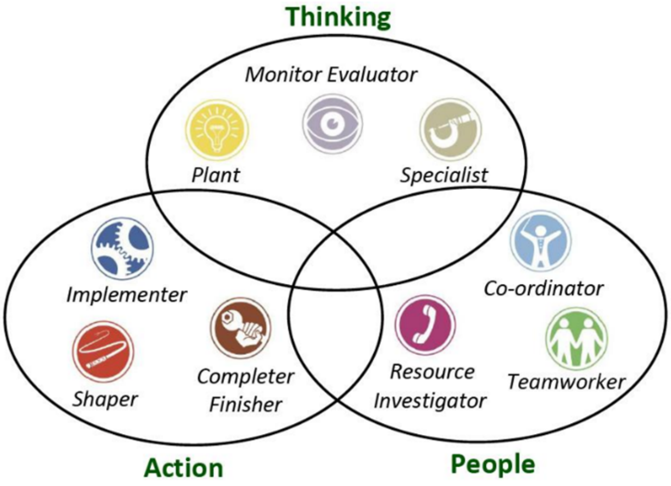


Figure 10: Divisions of Belbin team roles.

Belbin offers an intelligence people is needed to form a good team where the CO or the PL is the best. The best team contains an expert which is more likely to be a CH(CO), PL or ME, a leader who might be a CH(CO) or a TW and a PL or ME who corrects the expert peoples. Besides that Omar et al. (2016) considers only SH and PL to be the required roles for a SW team.Different team role has different strength. While PL and SP could be the creative part of the team (Mostert 2015) CH/CO and SH can be the leader. Moreover, an extrovert and an introvert people are required for a good team. He also warns up to avoid bad groups (Table 8).

Table 8: Bad team formations based on Belbin (1981)

Formation	Reason
1 CH (CO) and 2 SH	CH (CO) cannot lead the team
2 PL	Creativity is not realized
1 ME + TW and CW without PL	Strategy is not incorporated into the work
CW-s without PL or RI	No leading
TW + CW + CF without others	Failure detection is not realized
(PL + CW-s) + SH	SH tense the team up
PL + RI without others	No coordination
CW + CF + ME without others	Slow team go in details

It is important to note that Belbin has a different view of roles than the traditional role theory. He believes that team members have two types of roles. The first type is a typical functional role, as described as part of role theory. The second type is a team role, which comes from the set of roles described in the previous paragraph. Investigating how these types of roles affect team performance is germane this investigation. For a particular individual, the functional type of role might be a typist on a programming team, whereas the second type might be a company worker and a team worker. Individual members can fill more than one team role. The team role describes how the individual fits into the team, not what particular function he or she performs.

2.4.5 Team synergy

As the Section 2.4.1 presented, many factors can influence teamwork as a source of diversity. Diversity can lead synergy effect (Dwertmann et al. 2016) but synergy does not arise by itself within a group because of diversity, although diversity is obviously necessary for the development of synergy (Van Dijk et al. 2012).

According to the Cambridge dictionary the definition of **synergy** is as *the combined power of a group of things when they are working together that is greater than the total power achieved by each working separately*.

Synergy is usually used to describe the value of the interactions between the team members, so that how well the team members work together (Liemhetcharat and Veloso 2012). Since in a diverse team the members can contribute to the team's success in different ways, so it is natural that this appears as synergy in complex tasks. The effectiveness of a team in completing a task relies not only on the individual capabilities of its members but also on the synergy created through their collaboration and the creation of the team. A team's success is shaped by both the strengths each member brings and the harmonious interplay between them (Liemhetcharat and Veloso 2014). Thus, synergy describes how team members can work together rather than an individual trait. Since the effectiveness of cooperation can be both positive and negative, synergy can be both positive and negative. Although there were already examples of the description of the synergy network in the literature (Liemhetcharat and Veloso 2014), it was Kosztyán et al. (2022) who was the first to take into account the possibility of negative synergy and provided a method for scheduling projects taking synergistic effects into account. Although the authors also complain about the synergy measurement. This is indeed a difficult task, as not only the many characteristics of the individual can affect team synergy, but also the quality of interaction with other team members and the structure of the team itself. Although there are attempts to measure synergy (e.g. Fandel et al. (2012)), it can only be measured afterwards by re-measuring a teamwork (Hertel 2011). Therefore, instead of measuring synergy, an estimate of synergy is used, which is called synergy potential (Muniz and Flamand 2023, Hess 2022, Mumford and Mattson 2009)

Since I distinguish between the synergy network (measuring joint work) and the sociogram (measuring social relations), I can conclude that balanced DISC groups strongly influence the synergy network among team members. Based on Scullard and Baum (2015), I estimated the positive and negative synergy potentials between DISC behavioral types in a diverse group. The sociometric analysis also shows only positive relations due to the team's heterogeneous behavior, but in this thesis, I focus solely on the project's benefits, represented by the synergy network. Figure 11 shows positive synergies between D and I, D and C, I and S, and S and C, while negative synergies exist between D and S and I and C.

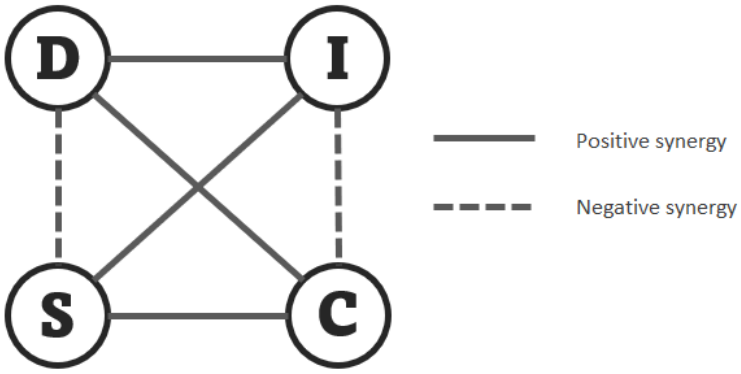


Figure 11: The assumed synergy network between DISC behavioral types

The synergy network revealed positive pairwise synergies between D and I, D and C, I and S, and S and C and negative pairwise synergies between D and S and between D and C. The details are shown in Table 9

Table 9: Synergies of pairs

(a) Pairs with positive synergy

First type	Second type	Benefit from the first type	Benefit from the second type
D	I	leadership and direction	inspire and motivate others
D	C	*focus on achieving results*	*attention to detail*
S	I	stability and consistency	build relationships
S	C	work collaboratively	attention to detail

(b) Pairs with negative synergy

First type	Second type	Loss from the first type	Loss from the second type
D	S	delegate and make pressure	inflexible
C	I	standoffish	open to discuss

Similar to the DISC the estimated synergy network can be created among the Belbin's team roles and the synergy matrix is attached Figure 12 based on many researches (Belbin 1981, Twardochleb 2017, Rajendran 2005, Monsalves et al. 2023)

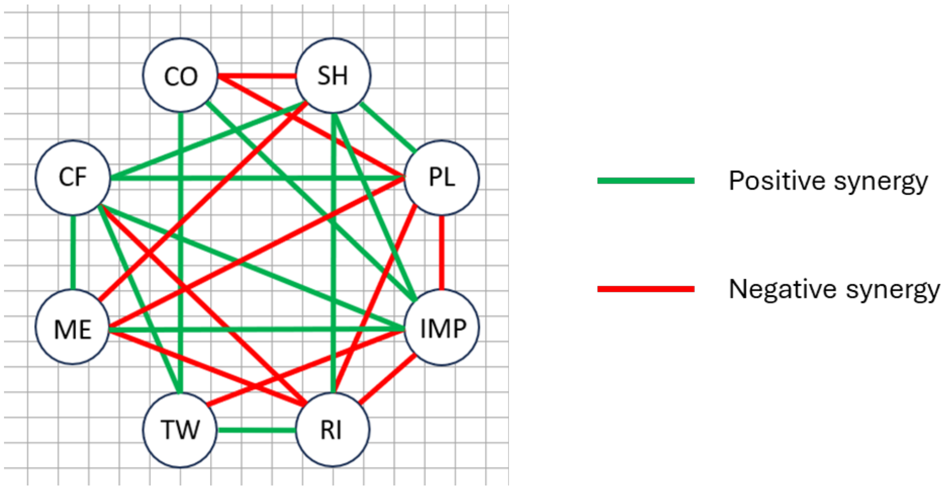


Figure 12: Estimated synergy network between Belbin team roles. Source: own illustration

If I disassemble the synergy network of Belbin's team roles into positive and negative synergy networks, I get Figure 13. In this figure, I can see that the action-oriented group plays a central role in the positive synergy network. The members of this group not only have the most positive relationships, but indirectly create the relationship between the members of the people-oriented and thinking-oriented groups.

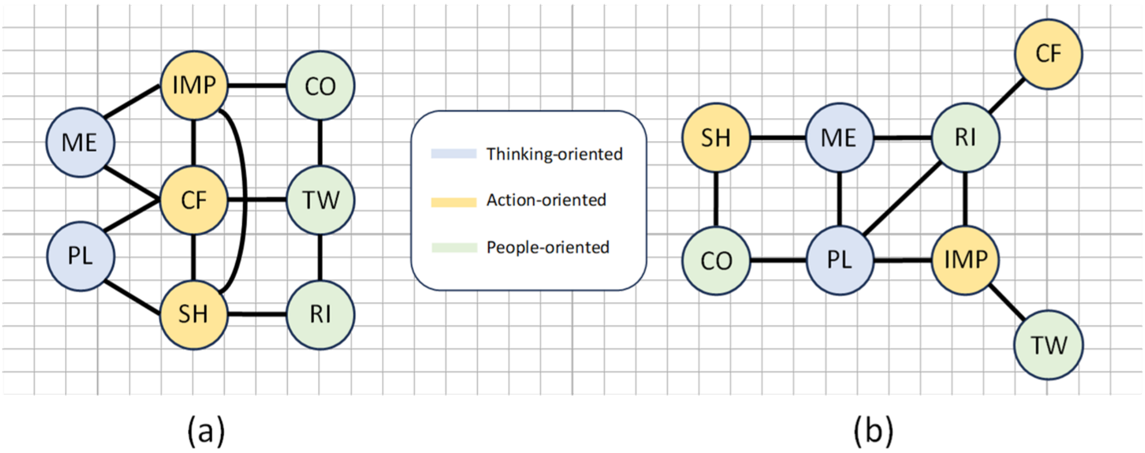


Figure 13: Positive (a) and negative (b) synergistic relationships between Belbin's team roles. Source: own illustration

2.4.6 Team selection aspects

Based on the literature reviewed so far, I can say that the software project team has the biggest impact on the success of the software projects. The experience and the leadership style of the software project team is strongly correlated with the success of the software project (Garousi et al. 2019). Besides that Wu et al. (2017) identified that the relationship conflicts can impact on the success negatively while the task conflicts influences the success positively. Conflicts and collaborations within the software project teams depend on the social skills of the team members (Lee et al. 2015)

In addition to being able to respond to continuous changes with agile methods, this approach places greater emphasis on teamwork, cooperation and personality (Kazemifard et al. 2011). Zainal et al. (2020) summarized the success criteria for the agile software development (ASD) team. They described that the incompatible personalities or behaviors, the imbalanced team roles and insufficient skill sets in a team can cause ineffectiveness which leads by the inappropriate team formation. The authors proposed a conceptual model for ASD team formation (Figure 14), where they separated the factors influencing the characteristics of agile team into six different fields which are influenced by the characteristics of the project to be done.

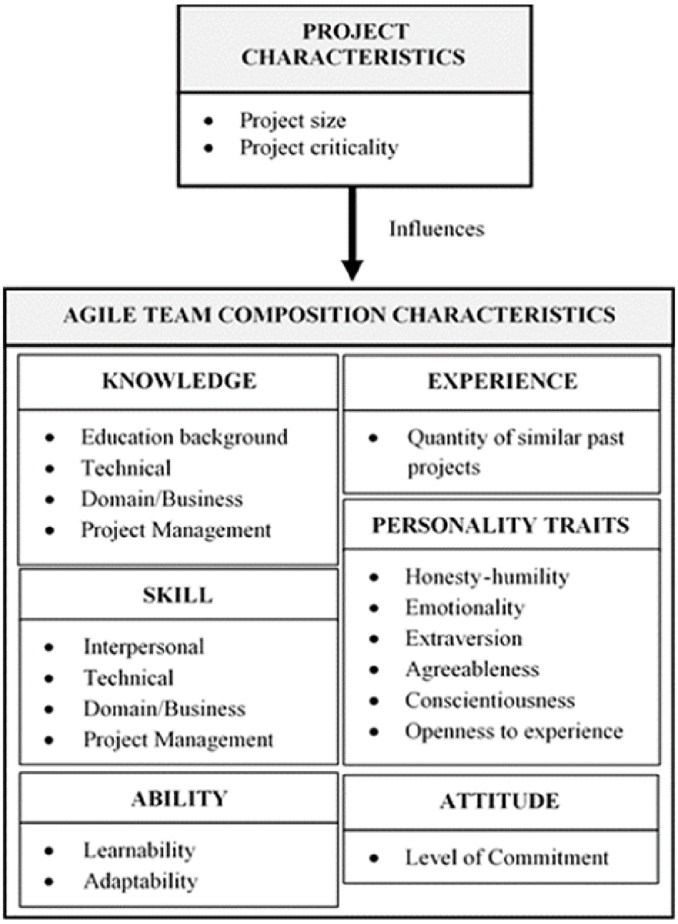


Figure 14: Concept of agile software team formation, based on Zainal et al. (2020)

As project management approaches were invented for projects of different sizes and complexities therefore, it is needed to distinguish between the teams that execute these projects. Main differences are summarized in the Table 10. In the past, organizations often preferred a hierarchical structure, with teams organized around specific functionalities. However, agile methodologies emphasize self-organization, fostering cross-functionality within teams. This approach allows individuals with diverse skill sets to come together autonomously, without a designated leader or external directive. While feature teams are typically large, self-organized teams are generally smaller to minimize complexity. As the number of team members grows, the increase in connections can lead to communication challenges and interpersonal issues. Given the high complexity of modern projects, team formation today often follows a hybrid project management approach. This method combines the strengths of both traditional and agile frameworks, adapting to the unique requirements of each project, which can be seen as temporary organizations. Regarding the team size I can conclude that the teams driven by agile approaches require less effort than the waterfall driven teams. However agile approaches can be used in small size projects which need less effort. The researchers (Bustamante and Sawhney 2011) offer 5 ± 2 for the agile development teams while for the waterfall teams it is suggested to be 25-75 or more, based on the size of the project.

Table 10: Differences between project teams based on their project management approaches

	Traditional	**Agile**	**Hybrid**
Team organization	Hierarchically organized team	Self-organized team	Hybrid team
Team functionality	Functionality-based team	Cross-functional team	Hybrid team
Team size	Large	Small	Medium
Team location	Anywhere	Preferred in one location	Hybrid
Leadership	Function-based leaders	Leaders are the part of the team	Highly collaborated and experts

Cross-functional teams can be formed with different basis. Using different team roles or behavioral types to select a team is called personality-based selection, and the team is the so-called synergistic team (Belbin 2012). This selection method involves selecting team members based on their personal traits with the goal of building a team that can collaborate effectively. This method can be effective when working on projects that require a high degree of communication and teamwork, or when working on projects with tight deadlines. If I do so by considering only the skill of the employees, I can call it a skills-based selection, and the formed team is the so-called multi-skilled team (Kaliprasad 2005). Multi-skilled team in regards of cognitive diversion can perform better in creative environment due to the centralization of many capabilities, but coordination of the team is required and payable. Moreover when responsibilities or requirements change the multi-skilled team usually cannot handle this well due to the difficulty of coordination (Lix et al. 2022). Either or they mixture, called synergistic multi skilled team could be an agile team. However, based on my knowledge I do not know of any method that tells which is better in an agile environment. If both personal traits and skills are considered during team selection it is called hybrid selection. This method can be effective for complex projects that require a high degree of technical expertise and collaboration. But in practice, where it is difficult to predict its future properties due to the project's characteristics (typically in the case of research and development projects for which neither the requirements nor the final product are known), the selection can be unique, and this is called random team selection.

Based on Meslec and Curşeu (2015) heterogeneous personalities has a greater chance of effective learning due to synergistic effects but not in later phases, meanwhile team role balance positively predicts group cognitive complexity and is negatively related to teamwork quality. However according to Batenburg et al. (2013) there is no relation between team role diversity and team performance. Considering personality profiles when forming teams has been shown to enhance performance, collaboration, and knowledge acquisition in student courses (Capretz and Ahmed 2010). The self-organized team – as a synonym of autonomous team – can be classified in skills-based selection group because this team is often formed by the employees' hard skills (Hoda et al. 2010). It is supported by Takeuchi and Nonaka (1986) and Beck et al. (2001) in flexible and complex environment. The well-known self-organized team was recognized before the agile methodology. Moreover, the problem-solving capability is very high of these teams (Tata and Prasad 2004), which can be exploited through the innovation (Takeuchi and Nonaka 1986). Moreover, under uncertainty, self-organized software development teams can maintain a sufficient autonomy (Dingsøyr and Dybå 2012). There is no leader or manager who distribute the tasks and monitoring the daily work, leadership is the responsibility of the team itself (Beck et al. 2001, Hoda et al. 2010).

In terms of hybrid teams, these teams do not have specific characteristics, because it depends on how agile the team is. However, I need to mention that define the metrics of agility is difficult due to its complexity. At this point I can use the attributes of the hybrid methods in Table 11, and classify the different kind of hybrid methods, presented by Reiff and Schlegel (2022).

Table 11: Properties of the hybrid teams

	Water-scrum-fall	**Waterfall-agile**	**Hybrid-V-model**	**Agile-Stage-Gate**
Team Organization	High skilled developers with traditional PM and testing	Closely cooperative developers and testers with traditional PM	Many-sided developers and continuous testing with high-skilled PM	High-skilled PM in cooperation with a many-sided scrum team
Usage	Companies taking their first steps towards agile often make only the development phase iterative.	In environments where stakeholders prefer a clear timeline but also want to incorporate feedback during development.	Ideal for projects where quality is critical, such as in healthcare or aerospace, and where requirements may evolve.	Effective for product development projects that require both innovation and a clear path to market, balancing flexibility with governance.
Shared leadership	low	medium	high	very high
Cross-functionality	low	medium	high	very high
Team size	medium	medium	medium	medium

2.5 Summary of the literature

During the last 30 years, project management has changed to different extents in various areas of the industry, thanks to the continuously changing world, where the complexity of projects and the competition between companies are increasing (see Section 2.3). Since the economic world is struggling with the management of restrictions, scarce resources and crises, it is essential to optimize projects for the implementation of companies' strategies.

This is especially important in the field of software development, which is increasingly important due to increasing digitization. Since development engineers are the most important resource in software development, so here project management was clearly drawn towards agile trends. However, due to the risks inherent in agility, various hybrid approaches are now widespread. In this environment, the optimal scheduling of projects is a complex scheduling problem, during which it is no longer enough to consider the success criteria according to the iron triangle, but emphasis must be placed on the efficiency of processes and customer relations. At the same time, retaining experienced people is also a strategic goal. That is why we can achieve the best output during project scheduling if we take into account the characteristics of the project team as much as possible. This is a team selection process, where the type of project, resource requirements, limitations and tasks determine the effective selection process. According to the type of project management, we can talk about traditional, agile and hybrid teams, where hybrid teams can carry traditional and agile characteristics to varying degrees. However, during this kind of scheduling, projects are still optimized according to the iron triangle.

Since the unity of the team is given by the developers with different abilities, it is not enough to consider only the performance of the team, but also the diversity of the developers and the interactions between different people must be taken into account when scheduling projects. These differences are mainly due to the type of skills and levels of skills of the developers, as well as their personalities and behavioral types. While we can talk about functional or cross-functional teams based on the levels and types of abilities and personality types or behavioral types. Accordingly, cross-functional teams can be skill-based, personality-based, hybrid or randomly selected teams.

Interactions between team members can be determined based on the effectiveness of the team members' common work, the synergistic effects between them. While the properties of skills and abilities and personality traits can be determined using already known methods, it is often not possible to quantify synergy. That is why the synergy potentials caused by differences can be used in practice. In the case of this dissertation, these differences are influenced by personality traits. Naturally, the nature of the project dictates the kind of people needed for its successful implementation. Although, the right people must be assigned to the right project tasks, but also the team must be selected during the scheduling of software projects. Therefore, taking into account team dynamics during the scheduling of software projects is also a kind of team selection process.

Thus, in addition to cross-functionality being realized, team dynamics must also be taken into account during optimization. When examined at the level of individuals, different types of people contribute to team dynamics in different ways and have different strengths in problem solving. Therefore, during scheduling, not only the existence of expert team members is important, but also the central team members who keep the team dynamics network (or synergy network in the dissertation) together. These central team roles can be identified with Belbin's team role theory, where action-oriented team members play a central role. However, the impact of these team members on the scheduling of software projects is still unknown. The impact of such central roles can be minimized if a smaller team is chosen. With 4 people, for example, cliques are less likely to form. During project scheduling, we can select a small team, or we can leave autonomy and schedule with an autonomous team as well. Practice shows that autonomous teams are able to cooperate more effectively and solve a specific task in the short term. Thus, the use of these teams can be beneficial for flexible and small projects, which is also preferred by agile project management. However, the reason for this is still little known.

2.6 Research assumptions

Drawing upon the findings within the literature, I have formulated the following three research assumptions (RA_1, RA_2, RA_3) to align with the research questions (RQ_1, RQ_2, RQ_3)

RA$_1$ The SSPSP method can be expanded to incorporate Belbin team roles and DISC behavioral types by leveraging the synergies among these roles, as well as the soft and hard skills they represent, within a flexible software environment.

RA$_2$ The presence of central team roles in software projects positively impacts project success, thereby enhancing performance within the constraints and objective functions defined in the supplemented SSPSP.

RA$_3$ Autonomous teams positively impact the success of software projects, within the constraints and objective functions of the enhanced SSPSP, more effectively than teams with dedicated leaders.

3 Method

The SSPSP specifies a flexible project plan modeled in a MDM matrix. Kosztyán et al. (2022) proposed a six-domain matrix model. The proposed version of the SMM matrix contains six domains and two column vectors. The modified SMM matrix is an $m + n \times m + s + n + 1$ matrix, where the number of employees is m, the number of skilled performances is s, and the number of tasks is n.

1. The first domain ($\mathbf{Y}$), an m by m submatrix, represents the synergy between employees. In the case of $i \neq j$

   - $[\mathbf{Y}]_{ij} > 1$ represents positive (or favorable) synergy,
   - $[\mathbf{Y}]_{ij} = 1$ represents neutral synergy,
   - $0 < [\mathbf{Y}]_{ij} < 1$ represents negative (or unfavorable) synergy between employees i and j.

   Furthermore, it is assumed that $[\mathbf{Y}]_{ii} = 1$ and that $[\mathbf{Y}]_{ij} = [\mathbf{Y}]_{ji}$.

2. The second domain is the skill domain ($\mathbf{S}$). The skill domain is an m by s submatrix, where every skill or skill performance is a nonnegative number ($[\mathbf{S}]_{ij} \in \mathbb{R}_0^+$). $s_j := s_{\cdot j} := \left[[\mathbf{S}]_{1j}, [\mathbf{S}]_{2j}, \dots, [\mathbf{S}]_{mj}\right]$ represent skill vectors. It is supposed that there are h **hard** skills, represented by the $[s_1, s_2, \dots, s_h]$ hard skill performance vector, and $s - h$ **soft** skills, represented by the $[s_{h+1}, \dots, s_s]$ skill vector. $[\mathbf{S}]_{ij} = 0$ means that employee i has no work ability for skill j. $\mathbf{S}$ can store either *binary* or *cardinal* levels of skills. In the case of the binary representation, $[\mathbf{S}]_{ij}$ is either 0 or 1. If the levels of skills (e.g., level of language skill, level of communication skill, etc.) are represented, then $[\mathbf{S}]_{ij} \in \mathbb{N}$. These binary and ordinal skills are referred to as *nonadditive*; in other words, there is no meaning of the sum $\sum_i [\mathbf{S}]_{ij}$. The cardinal skill performance (usually hard skill performance) is *additive*, and $[\mathbf{S}]_{ij} \in \mathbb{R}_0^+$. Synergistic factors may also modify skill performance. Let ε be a subset of employees; then, the *joint skill* of ε is

$$S_j^{\varepsilon} := \overline{Y}_{\varepsilon} \cdot \sum_{i \in \varepsilon} [\mathbf{S}]_{ij} \tag{1}$$

   where $\overline{Y}_{\varepsilon}$ is the *geometric mean* of synergies:

$$\overline{Y}_{\varepsilon} := \begin{cases} 1 & \text{if} \quad |\varepsilon| \leq 1 \\ \sqrt[\eta]{\prod\limits_{i,j \in \varepsilon,\, i<j} [\mathbf{Y}]_{i,j}} \qquad \text{where } \eta = \frac{|\varepsilon| \cdot (|\varepsilon| - 1)}{2} & \text{if} \quad |\varepsilon| > 1 \end{cases} \tag{2}$$

3. The third domain is a matching domain (**M**). **M** is an m by n domain, where $[\mathbf{M}]_{ij} \in [0,1]$ represents the maximal relative amount of assignments of employee i to task j. If $[\mathbf{M}]_{ij} = 0$ ($[\mathbf{M}]_{ij} = 1$), then employee i is not (fully) assigned to task j.

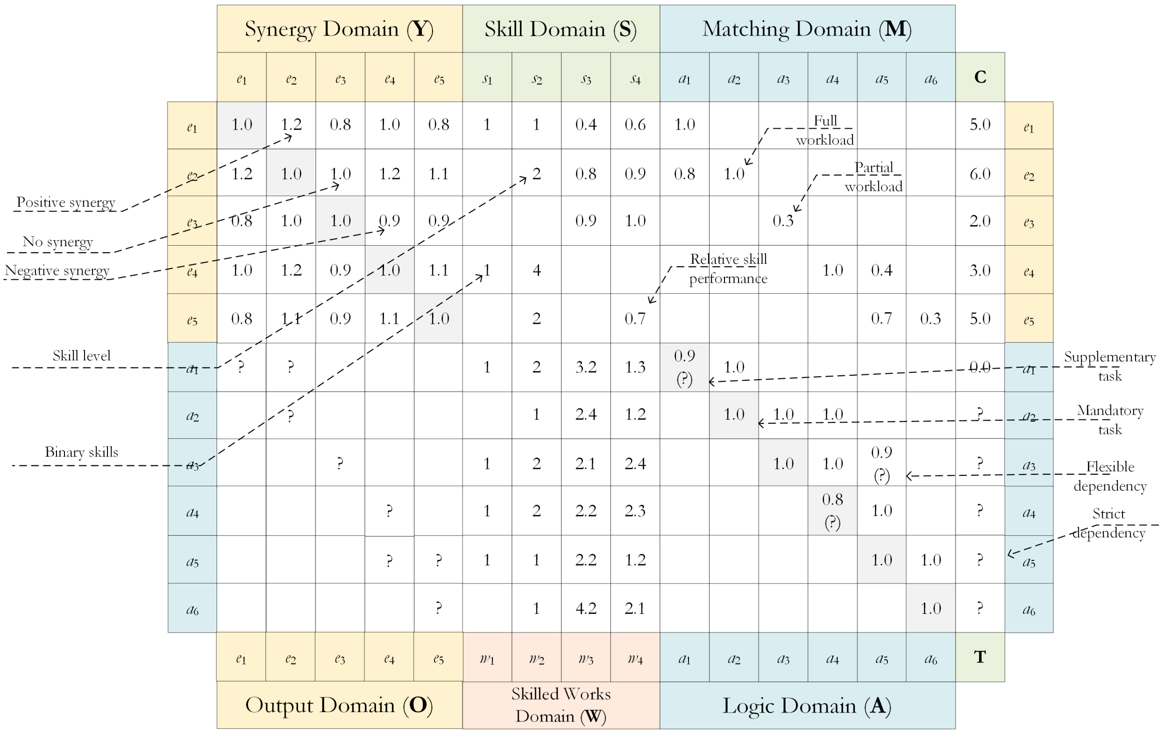
4. The fourth domain is the activity or task domain (**A**). The activity domain is an n by n square matrix ($[\mathbf{A}]_{ij} \in [0,1]$), where the diagonal represents the relative priorities of task (activity) completion. $[\mathbf{A}]_{ii} = 1$ represents the *mandatory* task, which must be performed ultimately and cannot be postponed. $0 < [\mathbf{A}]_{ii} < 1$ represents the *supplementary* task, which, depending on the constraints, can be postponed to a later project (or a later subproject, called a sprint in agile project management). A higher $[\mathbf{A}]_{ii}$ on the diagonal represents a greater priority (greater score). A postponed task's precedence and demands are also neglected. $[\mathbf{A}]_{ij}, i \neq j$ represents the precedence between tasks a_i and a_j ("a_i must end before a_j starts" ($a_i \prec a_j$), or "no precedence between a_i and a_j" ($a_i \sim a_j$)). $[\mathbf{A}]_{ij} = 1$ represents the fixed dependency between task i and task j ($a_i \prec a_j$), while $0 < [\mathbf{A}]_{ij} < 1$ represents flexible dependency ("$a_i \bowtie a_j$"), which, depending on the constraints, can be either prescribed or relaxed. Importantly, after the optimization, depending on the constraints, every $0 < [\mathbf{A}]_{ii} < 1$ and $0 < [\mathbf{A}]_{ij} < 1$ value must be either 1 or 0, and $[\mathbf{A}]_{ij} = 1$ must imply $[\mathbf{A}]_{ii} = 1$ and $[\mathbf{A}]_{jj} = 1$. This means that the proposed algorithm has to decide which supplementary tasks have to be completed or postponed and which flexible tasks have to be prescribed or relaxed.

5. The fifth domain is the skilled-word domain (**W**). **W** is the n by s matrix. $[\mathbf{W}]_{ji}$ stores the required skilled work of skill i for task j. In the case of binary and ordinal skills, $[\mathbf{W}]_{ji}$ can represent a minimum requirement of skills for completing tasks j, e.g., minimum level of communication and minimum level of language skills, while in the case of skill performance, $[\mathbf{W}]_{ji}$ represents a minimal amount of skilled work, such as tested functions and documented programming codes.

6. The last domain is the output domain (**O**), which contains the solution of the SSPSP algorithm. **O** is an n by m matrix (of nonnegative real numbers), where the element $[\mathbf{O}]_{j,i} > 0$ represents the (final) allocation of employee i to task j. $[\mathbf{O}]_{j,i}$ is the proposed *ratio* of the working time e_i is allocated to a_j; clearly, $[\mathbf{O}]_{j,i} = 0$ means *no* allocation. $[\mathbf{O}]_{j,i} \leq [\mathbf{M}]_{i,j}$ and $\sum_{j=1}^{n} [\mathbf{O}]_{j,i} \leq 1$ must hold for each $j = 1, 2, ..., n$ and $i = 1, \ldots, m$, while $\sum_{j=1}^{n} [\mathbf{M}]_{i,j} \leq 1$ are *not* required for any $i = 1, \ldots, m$.

7-8. The modified SMM matrix contains two extra column vectors. **C**, the first is an m by 1 column vector containing the salary of employees, and the second, **T** is an n by 1 column vector containing the scheduled start time of the tasks.

Figure 15 shows an example of a filled modified SMM matrix, where the number of employees is five and the number of tasks is six (four mandatory tasks, two supplementary tasks). The example represents one binary, one ordinal skill and two skill performances. The symbol “?” represents the variable cells that must be optimized with the proposed algorithm.



	Synergy Domain (**Y**)					Skill Domain (**S**)				Matching Domain (**M**)							
	e_1	e_2	e_3	e_4	e_5	s_1	s_2	s_3	s_4	a_1	a_2	a_3	a_4	a_5	a_6	**C**	
e_1	1.0	1.2	0.8	1.0	0.8	1	1	0.4	0.6	1.0						5.0	e_1
e_2	1.2	1.0	1.0	1.2	1.1		2	0.8	0.9	0.8	1.0					6.0	e_2
e_3	0.8	1.0	1.0	0.9	0.9			0.9	1.0			0.3				2.0	e_3
e_4	1.0	1.2	0.9	1.0	1.1	1	4						1.0	0.4		3.0	e_4
e_5	0.8	1.1	0.9	1.1	1.0		2		0.7					0.7	0.3	5.0	e_5
a_1	?	?				1	2	3.2	1.3	0.9 (?)	1.0					0.0	a_1
a_2		?					1	2.4	1.2		1.0	1.0	1.0			?	a_2
a_3			?			1	2	2.1	2.4			1.0	1.0	0.9 (?)		?	a_3
a_4				?		1	2	2.2	2.3				0.8 (?)	1.0		?	a_4
a_5				?	?	1	1	2.2	1.2					1.0	1.0	?	a_5
a_6					?		1	4.2	2.1						1.0	?	a_6
	e_1	e_2	e_3	e_4	e_5	w_1	w_2	w_3	w_4	a_1	a_2	a_3	a_4	a_5	a_6	**T**	
	Output Domain (**O**)					Skilled Works Domain (**W**)				Logic Domain (**A**)							

Figure 15: The proposed, modified SMM matrix

The *duration* of activity a_j is denoted by[1] $a_j^{dur}(\mathbf{O})$. (This depends on resources modified by the synergy factor, as calculated in Eqs. (5) and (6).) The *starting time* of a_j is $a_j^{start}(\mathbf{O})$, and the finishing time is[2] $a_j^{end}(\mathbf{O}) = a_j^{start}(\mathbf{O}) + a_j^{dur}(\mathbf{O})$ (see Eq. (7)). The *duration of the project* is denoted by p_{dur} or TPT, and its cost is p_{cost} or TPC. The monthly *salary* of employee e_i is denoted by e_i^{salary} or $[\mathbf{C}]_i$.

Since task a_j *requires* $[\mathbf{W}]_{j,k}$ skilled work, the *required time* (duration) to fulfill the requirement (skill) k of task j without synergies is:

$$a_{j,k}^{dur}(\mathbf{O}) = \frac{[\mathbf{W}]_{j,k}}{\sum\limits_{i=1}^{m}\left([\mathbf{S}]_{i,k} \cdot [\mathbf{O}]_{j,i}\right)}, \tag{3}$$

[1] From here "($\mathbf{O}$)" refers to the final output $\mathbf{O}$ of the algorithm.

[2] it is recalled that $a_i \prec a_j$ implies $a_i^{end}(\mathbf{O}) \leq a_j^{start}(\mathbf{O})$.

and the *adjusted required time* (with synergies) is:

$$a_{j,k}^{dur,adj}(\mathbf{O}) = \frac{[\mathbf{W}]_{j,k}}{\overline{Y}_{\varepsilon_j} \cdot \sum_{i=1}^{m} \left([\mathbf{S}]_{i,k} \cdot [\mathbf{O}]_{j,i}\right)} \tag{4}$$

where $\varepsilon_j := \left\{i : 0 < [\mathbf{O}]_{j,i}\right\}$ is the set of employees ultimately assigned to task j.

Without considering the synergies, the duration time of task j is:

$$a_j^{dur}(\mathbf{O}) = \max_{0<[\mathbf{W}]_{j,k}} \left\{a_{j,k}^{dur}(\mathbf{O})\right\} \tag{5}$$

and (with synergies):

$$\widetilde{a}_j^{dur}(\mathbf{O}) := a_j^{dur,adj}(\mathbf{O}) = \max_{0<[\mathbf{W}]_{j,k}} \left\{a_{j,k}^{dur,adj}(\mathbf{O})\right\}. \tag{6}$$

The durations are used to calculate the finish times of the activities $a_j^{end}(\mathbf{O}) = a_j^{start}(\mathbf{O}) + \widetilde{a}_j^{dur}(\mathbf{O})$, where

$$a_j^{start}(\mathbf{O}) \geq \begin{cases} 0 & \text{if } \nexists\, a_i \in A, a_i \prec a_j \\ \max\{a_i^{end}(\mathbf{O}) : a_i \prec a_j\} & \text{otherwise} \end{cases}. \tag{7}$$

a_i is **critical** if neither its starting nor ending time can be changed when optimizing the project time: $a_{j_1}^{end}(\mathbf{O}) = a_i^{start}(\mathbf{O})$ and $a_i^{end}(\mathbf{O}) = a_{j_2}^{start}(\mathbf{O})$ for some j_1, j_2 such that $a_{j_1} \prec a_i \prec a_{j_2}$. The "critical" status of an activity can be varied during the optimization algorithm [3].

The values calculated above enable us to calculate the duration of the project as follows:

$$\text{TPT}_{nosyn} := \max\{a_j^{end}(\mathbf{O}) : j = 1, \ldots, n\}. \tag{8}$$

$$\text{TPT}_{syn} := \max\{\widetilde{a}_j^{end}(\mathbf{O}) : j = 1, \ldots, n\}. \tag{9}$$

If all supplementary tasks are postponed and all flexible tasks are excluded (formally, $\mathbf{A}_{\min} = \lfloor \mathbf{A} \rfloor$), in the case of full assignments (formally, $\mathbf{O} = \mathbf{M}^T$), I obtain a minimum TPT ($\text{TPT}_{\min}$). However, in practice, this approach is usually not feasible, given the constraints. The maximal TPT is infinite if there is no assignment.

The cost of the project (TPC) can be calculated as the sum of the salaries of employees that are paid for their dedication to the project. Since positive synergy decreases and negative synergy increases the duration a_j^{dur} to $\widetilde{a}_j^{dur}$, the project cost can be calculated with and without the synergistic effect, obtaining TPC_{syn} and TPC_{nosyn}, respectively. Formally:

[3] and $a_j^{dur}(\mathbf{O})$ and $\widetilde{a}_j^{dur}(\mathbf{O})$ are minimized by the algorithm

$$\mathrm{TPC}_{syn} = \sum_{i=1}^{m}\sum_{j=1}^{n}([\mathbf{C}]_i \times [\mathbf{O}]_{j,i} \times \widetilde{a}_j^{dur}(\mathbf{O})), \tag{10}$$

$$\mathrm{TPC}_{nosyn} := \sum_{i=1}^{m}\sum_{j=1}^{n}([\mathbf{C}]_i \times [\mathbf{O}]_{j,i} \times a_j^{dur}(\mathbf{O})). \tag{11}$$

The maximal amount of costs $\mathrm{TPC}_{\max}$ occurs in the case of full assignment and if all supplementary tasks are decided to be completed. The minimum value is 0 if there is no assignment to any task.

The TPS is not influenced by the synergy. It depends only on the set of decided-to-complete tasks, denoted by $\mathcal{A}$.

$$\mathrm{TPS} := \sum_{i \in \mathcal{A}} [\mathbf{A}]_{ii} \tag{12}$$

$\mathrm{TPS}_{\min}$ ($\mathrm{TPS}_{\max}$) occurs if all supplementary tasks are postponed (completed).

3.1 Possible target functions

Now, the possible objective functions are declared that I seek to optimize *simultaneously* (in Eq. (16)) by using the following algorithm:

$$\mathrm{TPT} \to \min, \tag{13}$$

$$\mathrm{TPC} \to \min, \tag{14}$$

and

$$\mathrm{TPS} \to \max. \tag{15}$$

These objective (target) functions can be considered a multiobjective problem or a composite objective (target) function and can be specified as follows[4] (Here, C_s, C_p, C_c and C_t are given reasonable constants):

$$z := 1 - \sqrt[w]{\left(\frac{C_t - \mathrm{TPT}}{C_t - \mathrm{TPT}_{\min}}\right)^{w_t} * \left(\frac{C_c - \mathrm{TPC}}{\mathrm{TPC}_{\max}}\right)^{w_c} * \left(\frac{\mathrm{TPS} - C_s}{\mathrm{TPS}_{\max} - C_s}\right)^{w_s}} \to \min, \tag{16}$$

The positive numbers w, which represent weights, are derived from the preferences of the decision makers and can be expressed as the sum of w_t, w_c, and w_s positive numbers. The decision makers' preferences can be determined using the analytic hierarchy process (AHP), which relies on pairwise comparisons to assess the importance of target functions. In this

[4]. Instead of Pareto sets, the multiobjective functions (13), (14) and (15) are combined into the single function in (16).

investigation, equal weights are assigned to each variable, with $w_t = w_c = w_s = 1$. Pareto optimization is particularly advantageous when distinct outcomes are obtained with each goal function. Multiobjective optimization yields a Pareto front, which allows the decision maker to select the most appropriate solution. Simultaneously, if the outcomes produced by the objective functions exhibit negligible disparities, then multiobjective optimization is not justified.

I assume the constraints $\mathbf{CR}_1 - \mathbf{CR}_9$ below.

3.2 Constraints

Based on the literature (see e.g. Kia et al. 2016, Chen et al. 2017, Maghsoudlou et al. 2017, Wang et al. 2022, Kosztyán et al. 2022), I can state the following constraints:

CR₁ All required resources for the project are of human force type and are always available. The employment of each employee e_i in the *project* is not allowed to exceed its *maximum* value: $e_i^w := \sum_{j=1}^{n} [\mathbf{O}]_{j,i} \leq e_i^{maxw} := \sum_{j=1}^{n} [\mathbf{M}]_{i,j}$. Clearly, [5] $0 \leq e_i^w \leq 1$ by $\sum_{j=1}^{n} [\mathbf{O}]_{j,i} \leq 1$. In addition, each activity must be performed by at least one human resource.

CR₂: The set of skills that an activity requires must be a subset of the union of skills of the employees who perform this activity (for each activity a I have $\{s : [\mathbf{W}]_{a,s} > 0\} \subseteq \bigcup_{e \in \varepsilon_a} \{s : [\mathbf{S}]_{e,s} > 0\}$, where $\varepsilon_a \subseteq \{e_1, ..., e_m\}$ is the set of employees assigned to a).

Moreover, the "joint" level of skills of employees involved is at least the level required by the activity planned. (More precisely, for any activity a and skill s, such that $[\mathbf{W}]_{a,s} > 0$, I require (i) for *nonadditive* skill s, there are several $e \in \varepsilon_a$ such that $[\mathbf{S}]_{e,s} \geq [\mathbf{W}]_{a,s}$, (ii) for *additive* skill s, I need $S_s^{\varepsilon} \geq [\mathbf{W}]_{a,s}$.)

Finally, for each *critical* activity a and *soft* skill s required to a at the minimum level $w_{i,j}$ (i.e., $[\mathbf{W}]_{a,s} \geq w_{a,s}$), at least one employee $e \in \varepsilon_a$ must possess skill s at minimum level $w_{a,s}$ (i.e., $[\mathbf{S}]_{e,s} \geq w_{i,j}$ must hold).

There are further constraints in SSPSP to manage flexible projects and to specify the set of implemented tasks.

CR₃: The TPS must be greater than a specified (score) constraint C_s, formally: TPS> C_s.

The following six additional constraints are the constraints of the project plan:

[5] ; see the matching domain (**M**) in Fig. 15.

CR$_4$: Overwork is allowed up to a *certain level* (roughly: $E^w = \sum_{i=1}^{m} e_i^w \leq K^w$ for some constant K^w, with minor exceptions).

CR$_5$: The development costs include the total salaries of all employees assigned in projects, and TPC must be less than the cost constraint (C_c).

CR$_6$: The TPT must be less than the time constraint (C_t).

CR$_7$: Each task must be started and executed once without interruption. In addition, the manpower assigned to a skill of each task must perform the assigned skill continuously without interruption.

CR$_8$: All the workforces assigned to various skills of each activity should start their work concurrently.

CR$_9$: The staff is fixed throughout the entire development process.

Importantly, predefined precedences must be maintained. However, in a flexible project, flexible precedences can change. In addition, tasks can be excluded or postponed to a later subproject (usually called a sprint). The structure of the implementation is the output of the optimization.

3.3 Applied hybrid genetic algorithm

Since SPSP is NP-hard (Xiao et al. 2013a), which is a special case of SSPSP. Kosztyán et al. (2022) showed that SSPSP is also NP-hard. Kosztyán et al. (2022) also proposed a hybrid genetic algorithm for solving the SSPSP problem. The suggested HGA has two phases. In the first phase, a GA is employed to specify the set of completed tasks and the final precendences. In phase 2, a NMM method is used to refine SST to balance the resource demands. This algorithm, which was also implemented in MATLAB has been extended and parallelized. The original settings of HGA are published in Kosztyán et al. (2022); therefore, I focus only on the extensions and modifications.

A chromosome encodes a probable solution of SSPSP. The proposed modification of the original chromosome structure enables the selection of a synergy network from a pool. Synergy networks are influenced by team roles. *N* different team roles specify *N* different synergy networks. In addition, the proposed method distinguishes skills into binary and ordinal skills, which are usually soft skills, and additive skill performances, which usually describe the performance of hard skills. Finally, I organized the chromosomes into a multichromosome structure (see Figure 16 to decrease the computational time and to share the best chromosomes in parallel computations. The proposed chromosome structure has four parts.

The first element of the chromosome is the selected number of the synergy network. It is an ordinal value from 1 to N. The second part is a binary sequence of the decision outcome of completing supplementary tasks and flexible dependencies. The length of this part of the chromosome is $n_F = n_s + n_f$, where n_s is the number of supplementary tasks and n_f is the number of flexible dependencies. The third part of the chromosome encodes the assignment ratios from the output domain. These real values must be in the interval. The number of assignment ratios (n_A) is the number of nonzero elements from the match domain (M). The last part encodes the SST of tasks. The number of elements in this part is n. Therefore, the number of elements of a chromosome vector is $N_c = 1 + n_F + n_A + n$. All four parts of the chromosome are of a different type; therefore, different crossover, mutation, and selection mechanisms must be proposed for these parts. For the first two parts of the chromosome, a uniform crossover mechanism is used. However, the parents may be infeasible; therefore, I assume that the feasible parents' genes are ten times dominant. In other words, a gene is ten times more likely to originate from feasible parents than from infeasible parents. For the third and fourth (continuous) parts of the chromosome, an arithmetic crossover function is used. Since all chromosomes encode a solution to the modified SSPSP, the feasibility of a solution can be checked. Although it may take some time to assess feasibility, my findings indicate that increasing the number of feasible individuals for recombination leads to improved convergence without compromising the quality of the best solution. In the case of DoE, this parameter has also been tuned (see Section 3.4).

A two-step mutation process is used, in which the first step is general and is conducted for all parts of the chromosome. In the first step, the algorithm selects a fraction of the vector entries of an individual for mutation, where each entry has a probability of being mutated. According to the results, this rate is specified as 0.05. In the second step, although the same mechanism is used when the mutation operator is implemented, the two parts of the chromosomes must be distinguished. In this case, the adaptive feasible mutation function is used. The mutation operator chooses a direction and step length that satisfy the bounds and linear constraints. After the mutation operator is used, the requirements of the excluded tasks and their task dependencies must be eliminated (set to 0).

To reduce the computation time, I specified a multichromosome structure. Since the length of the chromosomes is equal in all runs, GA can be parallelized. In a multichromosome structure, there are p chromosome vectors. The value of p is usually related to the number of processors (or graphical co-processors). The pool contains M multichromosomes in a generation. The processors evaluate them in parallel, and selected chromosomes or parts of the chromosomes can be migrated (see Figure 16).

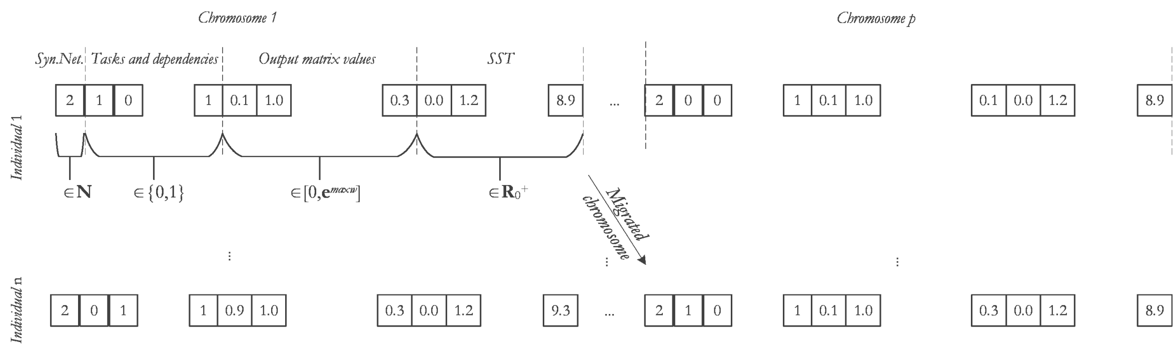


Figure 16: The proposed multichromosome structure

3.4 Hyperparameter tuning of GA

The changed SSPSP problem is more similar to the original SSPSP problem from Kosztyán et al. (2022), except for the formation of multichromosomes. I used the hyperparameters suggested there as a starting point because they were already tuned. I used the DoE method to adjust the hyperparameters of GA following Reeves and Wright (1999)'s theoretical background, and the optimization results were analyzed by a highly robust method, the so-called regression tree ensemble model of the MATLAB regression learner app. During the calculation, 10-fold cross-validation was used, and the hyperparameters were tuned via Bayesian optimization. The target function was the convergence speed. After tuning, the following parameters of GA are used (see Table 12).

Operators	Parameters
Population size	250
Badges of chromosomes	4
Turnament size	9
Elite count	0.05
Crossover fraction	0.82
Rate of feasible chromosomes in crossover	0.88
Probability of mutation of a gene in a chromosome	0.05
Maximal rate of migrated chromosomes	0.09
Tolerance value	$1E-8$
Maximal iteration	150

Table 12: Tuned hyperparameters of GA

After the hyperparameters of GA are tuned, the number of populations in a badge of chromosomes is set to 250, while the number of badges is 4. The number of badges depends on the core of the processors. Since the employed computers can run four independent threads in two cores, these values produced the best performance. The best selection mechanism was the tournament, for which the tournament size was 9. The elite count was 0.05. This value specifies how many individuals in the current generation are guaranteed to survive to the next generation. The crossover fraction specifies the fraction of each population (other than elite children) that consists of crossover children. After fine-tuning, 82% of the chromosomes were crossover children. The rate of feasible children was 88% in a crossover function. This value ensures fast convergence, but infeasible children help us not to consider only local minimum. The probability of a gene mutation was 5%, while the maximal rate of chromosome migration was 9%. The applied tolerance was 1E-8, while 150 iterations were always sufficient.

4 Experiments

Answering RQ_1 it is necessary to test the modified SSPSP on a validated database and compare the results with the results of validated methods. To initially test the modified SSPSP algorithm, the Imopse database was used (Myszkowski et al. 2019), due to its reliability (see in Section 2.2).

For the first trial I used the 15_6_10_9 project project from these databases, which contains 15 tasks, 6 employees, and 9 skills. However, this database does not use the following:

- *synergies*; therefore, the applied synergy matrix represents only no synergies: $([\mathbf{Y}]_{ij} := 1)$;
- *flexible dependencies*; therefore, only binary values in **A** is considered.
- *skill performance*; therefore, I considered skill as an ordinal variable, where the addition operator had no meaning.
- *non binary employee-task assignments*; only 0 or 1 values in domains **M** and **O** are allowed.

Table 13 shows the skill assignments (see Table 13(a)), the precedence matrix (see Table 13(b)), and the employee-task assignments (see Table 13(c)).

Table 13: Skill assignments and the precedence matrix.

(a) Skill assignments

	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9
e_1					1		2		2
e_2				1	2		1	1	
e_3	2					1	1	1	
e_4	2			2			2		2
e_5			1			2	2		
e_6	1	2							1

(b) Precedence matrix

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
a_1	1									1					
a_2		1	1	1											
a_3			1												
a_4				1						1					
a_5					1										
a_6						1	1								
a_7							1		1				1		
a_8								1							
a_9									1	1			1		
a_{10}										1				1	
a_{11}											1				
a_{12}												1			
a_{13}													1		
a_{14}														1	
a_{15}															1

(c) Task assignments

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
e_1		1				1						1			
e_2					1			1		1					
e_3	1										1		1		1
e_4			1	1											
e_5									1					1	
e_6							1								

Due to the restriction of one person per task, I restricted my evaluation to a maximum of one participant per task. After the optimization, the **T** vectors for the durations were **T**=[26.0006, 21.0008, 24.0001, 21.0004, 39.0000, 33.0004, 13.0014, 12.0000, 9.0003, 16.0002, 9.0001, 36.0004, 14.0000, 31.0000, 17.0000], where in the case of optimal assignments, the values of the duration times were $\mathbf{T}_{opt}$ =[26, 21, 21, 39, 33, 13, 12, 9, 16, 9, 36, 14, 31, 17]. After running the proposed HGA method, the TPC was 16279.1, while the optimal TPC was 16278.6. Although the results are very close to the validated real values, the real advantage of the proposed method is highlighted when synergies between employees are stated, not only are binary assignments considered, and the addition of hard skill performances is allowed.

Although the original test dataset does not specify synergies between employees, in the second step of the experiment, I considered the following synergy domains ($\mathbf{Y}_I$).

$$[\mathbf{Y}_I]_{ii} = 1,\ [\mathbf{Y}_I]_{ij} = y_I,\ y_I \in \{0.5, 0.6, \ldots, 1.5\},\ i \neq j. \tag{17}$$

Figure 17 shows the dependence of lead time and costs on the change in synergy. Considering Eqs. (8)-(9) and (10)-(11), I calculated TPT and TPC when considering and neglecting synergies. If the rate of $\text{TPT}_{\text{syn}}/\text{TPT}_{\text{nosyn}}$ ($\text{TPC}_{\text{syn}}/\text{TPC}_{\text{nosyn}}$) is lower than one, then one can see how the makespan (total cost) is reduced, while a value greater than one indicates the effect of the negative synergies.

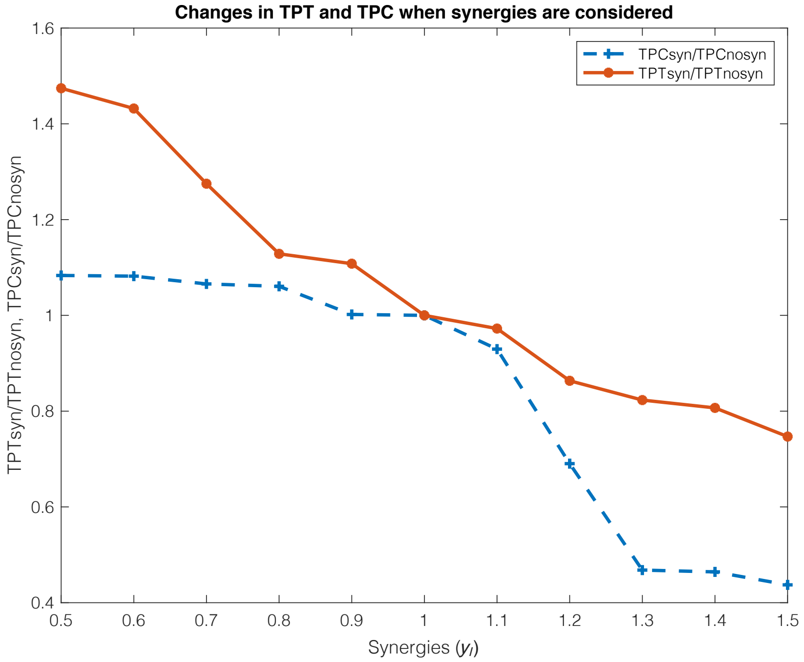


Figure 17: The dependency of total project time and total project cost on synergies

Since the synergy between all employees in this task changes simultaneously in the same direction, the lead times and costs are also sensitive to changes in the synergy. Figure 17 shows that costs ($\text{TPC}_{\text{syn}}/\text{TPC}_{\text{nosyn}}$) are more sensitive to positive synergies ($y_I > 1$), while lead time (see $\text{TPT}_{\text{syn}}/\text{TPT}_{\text{nosyn}}$) is more sensitive to negative synergies ($y_I < 1$).

(Myszkowski et al. 2015b) compared different methods for solving MS-RCPSP problems. Table 14 shows a comparison of the results. The first four algorithm results are from Myszkowski et al. (2015b)'s runs, and I use these results as a reference. These algorithms include (1) a simple duration-oriented heuristic algorithm, presented by Myszkowski et al. (2013); (2) a duration-oriented greedy algorithm; (3) the well-known metaheuristic ant colony algorithm (ACO); and (4) the modified hybrid ant colony algorithm (HAntCO), which can use priority rules against the simple ACO, where (2-3-4) are presented by Myszkowski et al. (2015a). The original SSPSP algorithm was implemented by Kosztyán et al. (2022), and the modified, poposed SSPSP algorithm was used as the HGA algorithm.

Table 14: Comparision of existing methods in MS-RCPSP (n is the number of tasks, e is the number of employees, p is the number of precences, s is the number of skills, TPT is the total project time, TPC is the total project cost. ACO is the Ant Colony Optimization, HAntCO is a the modified (heuristic) Ant Colony Optimization, SSPSP is the algorithm for the synergy-based software scheduling problem.

				Heuristic		Greedy		ACO		HAntCO		original SSPSP		modified SSPSP	
n	*e*	*p*	*s*	TPT	TPC	TPT	TPC	TPT	TPC	TPT	TPC	TPT	TPC	TPT	TPC
100	10	26	15	37	126361	38	119336	32	124687	31	126216	35	124168	35	124154
100	10	27	9	38	44309	38	43438	34	44999	33	42199	37	43756	36	43750
100	10	47	9	41	142759	40	135161	36	143100	34	140865	38	140483	38	140489
100	10	48	15	36	135534	44	120664	33	133062	33	133495	37	130698	37	130693
100	10	64	9	39	113124	43	117993	35	110643	33	113774	38	113898	38	113892
100	10	65	15	40	152955	43	140782	35	150294	32	149185	38	148305	38	148313
100	20	22	15	25	117493	24	112135	20	120949	19	123642	22	118568	23	118556
100	20	23	9	32	53154	32	50279	32	52119	23	53358	30	52235	31	52242
100	20	46	15	28	138270	29	133739	25	138565	24	138568	27	137286	27	137294
100	20	47	9	21	129160	28	140626	21	124817	18	134312	22	132235	22	132247
100	20	65	15	32	110503	34	118569	27	109831	27	108991	30	111987	31	111974
100	20	65	9	25	127149	24	124291	23	130934	21	126659	24	127267	23	127261
100	5	20	9	57	40539	55	40958	50	41029	53	40811	55	40841	55	40849
100	5	22	15	63	119266	77	128354	60	119434	60	119158	65	121570	65	121555
100	5	46	15	75	202238	80	202607	67	204110	67	204730	72	203422	73	203437
100	5	48	9	72	193383	78	196893	62	191712	62	191888	69	193471	69	193487
100	5	64	15	71	141407	66	141882	62	144972	61	143956	65	143068	65	143073
100	5	64	9	71	102439	67	107014	61	102777	61	101297	65	103385	66	103399
200	10	128	15	71	180812	78	198378	62	178264	60	178375	68	183960	68	183969
200	10	135	9	216	105593	216	93426	216	99375	186	103561	209	100508	209	100492
200	10	50	15	66	189660	75	183673	63	191856	62	190956	67	189042	67	189045
200	10	50	9	66	251158	70	250732	65	250075	64	250850	67	250721	67	250717
200	10	84	9	70	224121	66	222976	69	226666	66	222655	69	224121	68	224110
200	10	85	15	65	304277	68	301357	61	306949	62	302064	65	303677	64	303682
200	20	145	15	36	275983	46	277097	36	278199	35	272504	39	275947	39	275956
200	20	150	9	183	92821	183	95667	186	91461	177	92567	183	93146	183	93143
200	20	54	15	37	295786	41	290656	39	299993	34	298822	38	296334	39	296330
200	20	55	9	37	230150	37	232766	38	231094	36	223879	38	229484	38	229486
200	20	97	15	49	290399	69	346527	42	280951	42	277860	51	298948	51	298935
200	20	97	9	35	273378	43	282379	37	275819	35	278797	38	277608	38	277596
200	40	130	9	112	101879	112	90907	112	94488	108	104965	112	98066	112	98079
200	40	133	15	24	276456	23	279170	27	281933	24	279073	25	279167	25	279178
200	40	45	15	31	260738	32	269623	25	248717	23	256687	29	258946	28	258942
200	40	45	9	22	270758	23	276416	26	273632	25	270428	25	272819	24	272824
200	40	90	9	24	290028	20	294909	26	287694	24	298340	24	292752	24	292758
200	40	91	15	19	249909	35	250843	25	257927	23	241492	26	250059	26	250049
			Mean:	54.61	176498.58	57.69	178117.31	51.94	176197.97	49.39	176027.19	53.92	176719.25	53.83	176719.44

Table 14 shows that HAntCo outperforms all the other methods and that the SSPSP algorithms have similar performances. The performances of the SSPSP algorithms in minimizing TPT and TPC can be considered average. Moreover, the advantages of these methods unfold when flexible task dependencies, different types of skills, and synergies between employees are accounted for.

Based on the results of the comparison and the test on the validated databases the new SSPSP method is accurate in the appropriate level. But taking the limitations of the test database into account the new method should be evaluated on real life examples.

5 Case Study

Answering RQ_2 and RQ_3 and to confirm the results obtained in Section 4, on a validated database, by taking into account those listed in its limitations, a multiple qualitative case study (Yin 2009) with validation process is planned (see in Table 15. Multiple case studies are useful for testing theories and methods furthermore, it is a good way to verify the method presented in Section 3 by examining several aspects together (Yin 2009).

The multiple case study is similar to several single case studies running side by side. The main difference is that the multiple-case contains an additional chapter covering the cross-case analysis and results (Yin 2009). In addition to the examination of the developed method on the Imopse database, with the case study it is possible to consider synergies, flexible dependencies, skill performance and non-binary employee-task assignments.

The case studies are took place at Continental Automotive Hungary Ltd., a large automotive company in Hungary. Continental was founded in 1871 primarily for tire production. Then, in 1998, the production of brake systems begins, and thus the segment of Continental Automotive is established and at the same time Continental Automotive Hungary Ltd. was founded. Today the company employs more than 200,000 workers at 505 locations in 56 countries, half of whom work in the automotive sector. The automotive sector also provides half of Continental's revenue of more than €40 billion (Continental Annual Report 2023).

The employees included in the case study are engaged in software development in the company's research and development (R&D) area in Hungary. In an R&D environment, creativity and innovation are top priorities (Mostert 2015). This division comprises nearly 500 professionals in software development, dedicated to enhancing the brake systems of cutting-edge vehicles. Brake systems contain one of the most complicated software, so even in the case of software development at this company, millions of lines of code are developed.

5.1 Organization projects in context of software development

R&D sector of Continental primarily uses waterfall method as project management method, but the sprint appears in it as an iterative approach and there are more releases than increments during the project's life cycle. Thus, a quasi-hybrid project management approach characterizes the company's projects which is called iterative V model. In the end of each iteration, a release is expected. Together with the product, the individual features also develop and each new release builds on the previous one. At the company, this is called the maturity life cycle (MLC), and the development of features is called feature maturity, which opens up the opportunity for maturity-driven development (MDD).

Development of brake control unit has 3 major levels: function level (1), system level (2) and sub components level (3) which includes mechanic, electronic and software parts. According to the V-model of product development, product development begins with the analysis of incoming requirements at function level than these are refined to system requirements. The system-level requirements are then divided into sub-components and they are realized at the level of the individual components. After the software development, which is at the bottom of the V-model all features will be tested at different levels just the opposite of how the requirements were processed. Thus, the actual development takes place at all three levels, but essentially the software is the one that is integrated into the electronics as the lowest-level component, and then into the system together with it, which is basically the brake control unit.

The project organization of the company's R&D sector is a matrix-based project organization, where the direct managers of the various feature teams are members of the line management, but their professional managers on the project side are project managers. This means that project-side responsibility rests with project managers, while line management is responsible for human resources. In addition, the superiors of the project managers are also line managers. The relationship between the projects and the line management is made possible by the strategy-oriented project organization itself, because the income of the company's R&D sector is mainly made up of its projects. The joint responsibility at the R&D level meets with the R&D director, who reports directly to the project directorate and the heads of the departments consisting of the individual feature teams.

From Q3 2023, the R&D organization switched to a type of agile development called the scaled agile framework during the portfolio of one of its project owners. Here the R&D follow several agile project management approaches like SCRUM[6] methodology. There are dedicated scrum masters and product owners but since these roles have only existed for a short time, I did not distinguish them during the case study either. In addition, the feature teams still exist and the iterative V model has also remained, so this type of agile development can also be said to be a hybrid methodology. All of this happened because of project owner's expectations, who still cannot switch to agile development. In this case, the corporate project organization is also matrix-based, but as the professional leaders of each feature team, a separate Scrum Master and Product Owner are responsible according to the Scrum organizational structure. With this, the professional influence of the project managers on the project organization and their responsibility towards the project result also decreased, and relations with the project owner and the cooperation of the feature teams on the project side became their primary tasks. Because of this, they often no longer participate in the professional tasks, so during the professional solution of a problem, the product owner is already responsible for the result created by the responsible feature team.

Since product development consists of features, project teams are also made up of these feature teams. Thus, software engineers specialize in the feature that their team is working on. For this reason, cross-functionality can be considered low. Members of feature teams are usually assigned to projects based on skill levels. The more important a project or the

[6] SCRUM is not an abbreviation

greater the complexity of the project, the more experienced an engineer is assigned to it. Past experiences are also taken into account when project teams are combined but this common work is not quantified. Estimated time for each task of a project is calculated based on the past experiences but it is often underestimated or overestimated. And the cost of the project's tasks is the product of the estimated duration and the salary of the person in charge. Since the execution time for a project task is uncertain, the estimated project cost can be very far from the reality.

Two aspects have been examined:

- The first aspect was examined in the traditional project management environment typical of most projects of Continental. According to this, the members of the Belbin's team's action-oriented group, as the central group, were basically absent from the group, and then the effect of their joining on the outcome of the project was examined one by one (see RA_2).

- The second aspect looked at Continental's new, agile project management team, already assembled in advance. Here, based on the DISC behavioral types, the impact of the dedicated leader during selection and the purely autonomous team on the outcome of the project was examined (see RA_3)

However, the company places great emphasis on people's well-being and development. That is why a large amount of data is available on employees' personality assessments, skill levels and the effectiveness of their joint work. Different data sources can be seen in Table 16. Individual characteristics and data are contained in the following subsections: Section 5.2 and Section 5.3. The data is presented according to the same structure for both subsections: description of the project tasks, presentation of the team's characteristics and finally the presentation of the different synergy networks.

All data used for the simulation was collected of people who would participate in such a survey and meet the requirements:

- in the past six months, they participated in Belbin or DISC training and their various team roles or behavioral types were assessed there

- they have worked in the same project teams for the past 1 year

- they have worked together on at least 10 tasks in the past 1 year and

- their hard skills for the past 1 year can be extracted from the internal database

The soft skills needed by team members to perform the tasks were measured using a 10-point Likert scale, which was divided by ten to obtain normalized data. While soft skills usually depend on personality type or behavioral types, hard skill performance usually depends on former experiences. These hard skills of each team member were measured by using the average data of the last 1 year.

Table 15: Report of the case study

Timeline	Process activity	Extended explanation
October 2022 - March 2023	Formulate the theory	In this period, the new SSPSP model was completed and those two aspects were formulated based on the RA_2 and RA_3. Both aspects are based on the evaluation of the selection process using the new SSPSP method.
April 2023 - December 2023	Identify and analyze the case	With the detailed analysis of the literature, the cases were essentially predefined with RA_2 and RA_3. According to this, in this session, 8 employees of the company were examined during the first case, while 4 employees were examined during the second case. The data was collected according to Section 5.2 and Section 5.2 using different databases (see in Table 16), and then the individual cases were analyzed by simulation in the Matlab environment.
January 2024 - May 2024	Evaluate solutions	In this period, based on the analysis of the results, it was possible to establish the best solutions for each case, despite the limitations of the model (see Section 6)
June 2024 - August 2024	Validate and verify the results	To verify the results of the simulation based on real data, I applied the methodology of participant observation. For the first case, 47 and for the second case 20 software developers are included who agreed to participate in the research and met the requirement of having different team roles based on the Belbin's Self Perception Inventory test or for the second case, different DISC behavioral types based on DISC training. The second step was to build the trust between them and establish teamwork through a half-hour moderated discussion. The third step was the observation where teams were participated in a Marshmallow challenge game which took 40 minutes: 15 minutes for the introduction of the game, 15 minutes to build the tower and 10 minutes to measure the height of each towers. At the stage four, I collected feedback from each teams regarding they feelings and finally analyzed the given data (see Section 7)

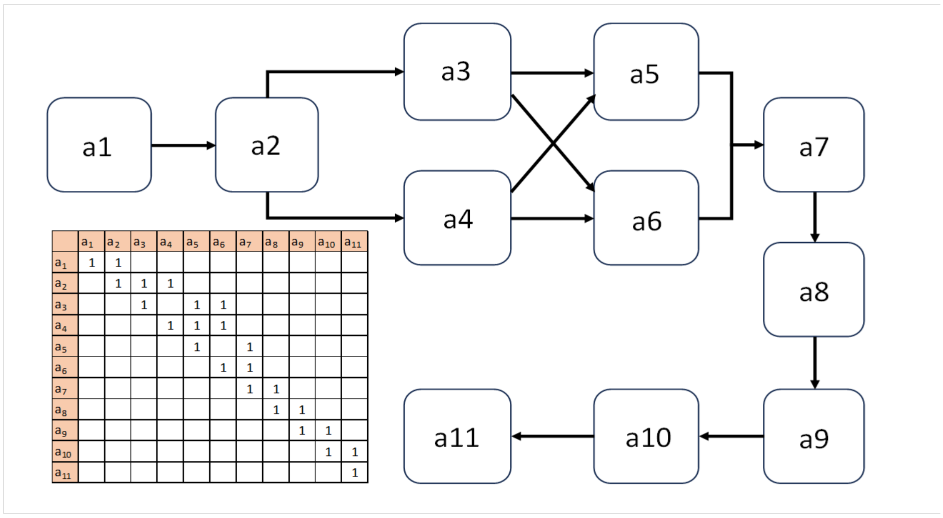
Table 16: Data sources

Data source	Details
Training material	8 training materials from the Belbin's training and 4 training materials from the DISC training were asked from the HR department to select the team members for the simulation where in both cases an external trainer made the results of the tests available. Additional 20 DISC training paper were asked for the validation process
Internal database	Internal database was used to measure hard skills in both cases
Questionnaire for the simulation	To measure the soft skills of the 8 different Belbin's team roles and 4 different DISC behavioral types a questionnaire was made by the HR department using 10 point Likert scale
Questionnaire of team roles	120 Belbin's self perception inventory questionnaire for determination of the Belbin's team roles of different project teams where 47 different Belbin's team roles were selected for the validation
Interviews	8 Belbin teams and 5 DISC teams were interviewed to report on their experience of validation teamwork

5.2 Data collection for central team roles

Collecting data for answering the RQ$_2$, to populate the SMM matrix for the simulation, I used a sprint of a real software development project with 11 tasks. I intentionally selected a sprint composed of tasks whose resource needs could be readily estimated from the experience with similar tasks in previous sprints, along with the interconnections between them. As a result of the sprint retrospectives, the estimation of the resource demand of the tasks could become more accurate.

The examined sprint consists of the following tasks: After the analysis of the requirements (a_1) and the refinement of the requirements (a_2) each feature needs to be implemented (a_3),(a_4) and tested (a_5)(a_6) separately, but in parallel, based on customer requirements. They plan a refinement of the implementation (a_7) to address any anticipated bugs identified during the testing phase. Software integration can be started when all functions have been successfully tested (a_8). Software integration is followed by the testing (a_9) of the integrated software, the preparation of the documentation for the process (a_10), and then the release of the software (a_11). All tasks and dependencies are required which is presented in Figure 18



	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}
a_1	1	1									
a_2		1	1	1							
a_3			1		1	1					
a_4				1	1	1					
a_5					1		1				
a_6						1	1				
a_7							1	1			
a_8								1	1		
a_9									1	1	
a_{10}										1	1
a_{11}											1

Figure 18: Logical structure of task precedences - Central Team Role Selection

The sprint has a duration of 2 weeks, so in order for the goals of the sprint to be fulfilled within the specified period, the requirements must be met in accordance with my estimate based on previous sprints.

To execute the sprint, 8 individuals were selected, each with distinct personality traits aligned with Belbin's team roles: the Plant(PL) (e_1), the Coordinator(CO) (e_2), the Team Worker(TW) (e_3), the Monitor Evaluator(ME) (e_4), the Implementer(IMP) (e_5), the Resource Investigator(RI) (e_6), the Completer Finisher(CF) (e_7) and the Shaper(SH) (e_8). They all took part in a Belbin's training, where the company identified their team roles with the Belbin Team Roles test, (validated by Witkowski and Ilski (2000)). The selected persons are software developers with an average of 5 years of experience, but at least 3 years. They have already known each other, they worked together before in one team. Required hard skills to complete the tasks are the written codes per week (s_1), the analyzed requirements per week (s_2), the written documents per week (s_3) and the tested codes per week (s_4). There are soft skills and abilities that can increase the speed at which tasks are performed. These are the communication ability (s_5), problem-solving skills (s_6), analytical thinking (s_7), collaboration skills (s_8) and the leadership ability (s_9). These skills and abilities are the main ones which have been using for many of other software development projects in this company. Table 17 contains all required skills of the Belbin's team.

Table 17: Skills versus team roles

(S) Skill	Soft skills					Hard skill performances			
Domain	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9
e_1 (PL)	0.50	1.25	2.00	0.50	1.00	200	35.0	-	1.00
e_2 (CO)	1.00	1.00	0.50	1.25	2.00	-	10.0	15.0	-
e_3 (TW)	1.25	1.00	1.00	2.00	1.00	-	-	5.00	-
e_4 (ME)	1.00	2.00	1.25	0.50	1.00	300	25.0	5.00	3.00
e_3 (IMP)	1.00	1.50	1.25	1.25	1.25	550	15.0	15.0	2.00
e_3 (RI)	2.00	1.00	1.50	1.25	0.50	-	35.0	-	-
e_3 (CF)	0.50	2.00	1.00	0.50	1.00	450	-	30.0	2.00
e_3 (SH)	1.25	0.50	1.00	1.00	2.00	-	10.0	-	-

Note. Central Team Role Selection.

Table 18 shows the skill-work requirements for each task.

Table 18: Skilled-work requirements for each task

	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9
a_1	1.00	1.25	1.75	1.00	1.25	-	50.0	-	-
a_2	1.25	1.50	1.25	1.75	1.50	-	180	40.0	-
a_3	1.00	1.50	1.25	1.50	1.25	200	12.0	3.00	-
a_4	1.50	1.75	1.25	1.50	1.25	400	24.0	3.00	-
a_5	1.00	1.25	1.50	1.25	1.00	-	24.0	2.00	2.00
a_6	1.00	1.25	1.50	1.25	1.00	-	24.0	2.00	2.00
a_7	1.50	1.50	1.00	1.00	1.25	100	-	30.0	-
a_8	1.25	1.25	1.50	1.25	1.00	-	10.0	2.00	1.00
a_9	1.25	1.00	1.25	1.50	1.50	-	-	-	-
a_{10}	1.25	1.00	1.75	1.50	1.25	-	20.0	100	-
a_{11}	1.75	1.50	1.00	1.75	1.75	-	-	-	-

Note. Central Team Role Selection.

Synergy network among team members were estimated according to the literature and based on their past common works, however, to maintain generality, in the simulations, ± 20% are added randomly to every positive and negative synergy value. Since the synergy matrix is symmetric and the diagonal values are 1.0, it is sufficient to specify the upper triangular part of the synergy matrix. I used 5 different values to describe the synergy network (Table 19. considering the ± 20% deviations, the modified values did not change the direction of the synergy (it was not allowed to change from negative synergy to positive synergy).

Table 19: Values used to describe synergy

type of the synergy	original value	maximum value	minimum value
negative	0.50	0.60	0.42
negative but approaching positive	0.75	0.90	0.63
no synergy	1.00	1.00	1.00
positive but approaching negative	1.25	1.50	1.05
positive	1.50	1.80	1.25

Note. Central Team Role Selection.

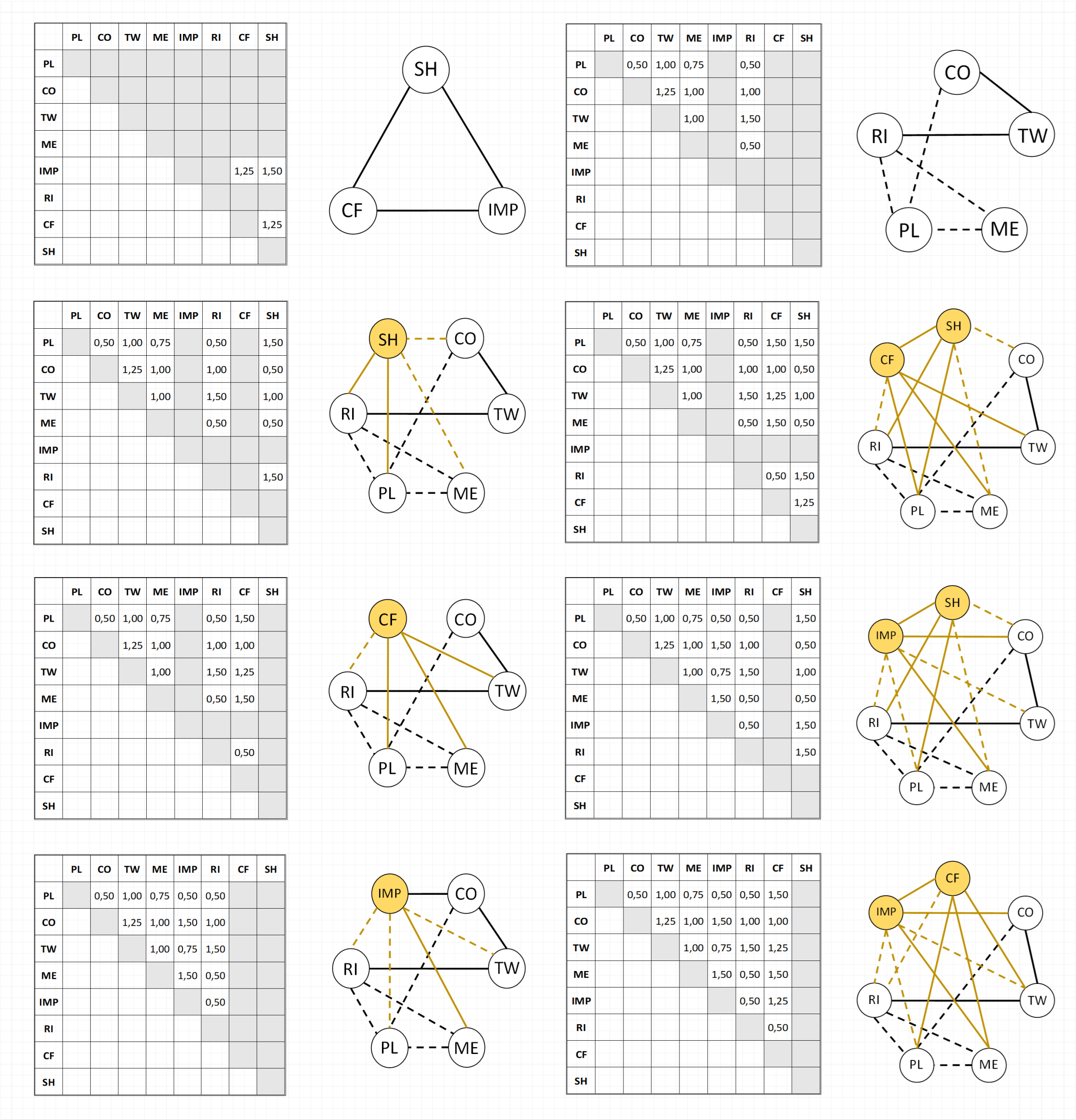
Using the positive/negative synergy values to describe the synergy between the Belbin's team roles, I get Table 20, where the 8 team roles are. In this case, for example, the value 0.5 (1.5) between two people means that the two people together can complete the given task in 1/0.5 = 2 times (1/2 = 0.5 times) as much time task compared to if they were working on it separately.

Table 20: Synergy matrix between employees whose team roles are categorized by the Belbin's team roles. (Only upper triangular are specified)

	e_1 (PL)	e_2 (CO)	e_3 (TW)	e_4 (ME)	e_5 (IMP)	e_6 (RI)	e_7 (CF)	e_8 (SH)
e_1 (PL)	-	0.50	1.00	0.75	0.50	0.50	1.50	1.50
e_2 (CO)		-	1.25	1.00	1.50	1.00	1.00	0.50
e_3 (TW)			-	1.00	0.75	1.50	1.25	1.00
e_4 (ME)				-	1.50	0.50	1.50	0.50
e_5 (IMP)					-	0.50	1.25	1.50
e_6 (RI)						-	0.50	1.50
e_7 (CF)							-	1.25
e_8 (SH)								-

Note. Central Team Role Selection. Only upper triangular is specified.

Finally I obtain eight different Belbin synergy networks where the central characters appear in different configurations. The generated synergy networks are presented in Figure 19.



	PL	CO	TW	ME	IMP	RI	CF	SH
PL								
CO								
TW								
ME								
IMP							1,25	1,50
RI								
CF								1,25
SH								

	PL	CO	TW	ME	IMP	RI	CF	SH
PL		0,50	1,00	0,75		0,50		
CO			1,25	1,00		1,00		
TW				1,00		1,50		
ME						0,50		
IMP								
RI								
CF								
SH								

	PL	CO	TW	ME	IMP	RI	CF	SH
PL		0,50	1,00	0,75		0,50		1,50
CO			1,25	1,00		1,00		0,50
TW				1,00		1,50		1,00
ME						0,50		0,50
IMP								
RI								1,50
CF								
SH								

	PL	CO	TW	ME	IMP	RI	CF	SH
PL		0,50	1,00	0,75		0,50	1,50	1,50
CO			1,25	1,00		1,00	1,00	0,50
TW				1,00		1,50	1,25	1,00
ME						0,50	1,50	0,50
IMP								
RI							0,50	1,50
CF								1,25
SH								

	PL	CO	TW	ME	IMP	RI	CF	SH
PL		0,50	1,00	0,75		0,50	1,50	
CO			1,25	1,00		1,00	1,00	
TW				1,00		1,50	1,25	
ME						0,50	1,50	
IMP								
RI							0,50	
CF								
SH								

	PL	CO	TW	ME	IMP	RI	CF	SH
PL		0,50	1,00	0,75	0,50	0,50		1,50
CO			1,25	1,00	1,50	1,00		0,50
TW				1,00	0,75	1,50		1,00
ME					1,50	0,50		0,50
IMP						0,50		1,50
RI								1,50
CF								
SH								

	PL	CO	TW	ME	IMP	RI	CF	SH
PL		0,50	1,00	0,75	0,50	0,50		
CO			1,25	1,00	1,50	1,00		
TW				1,00	0,75	1,50		
ME					1,50	0,50		
IMP						0,50		
RI								
CF								
SH								

	PL	CO	TW	ME	IMP	RI	CF	SH
PL		0,50	1,00	0,75	0,50	0,50	1,50	
CO			1,25	1,00	1,50	1,00	1,00	
TW				1,00	0,75	1,50	1,25	
ME					1,50	0,50	1,50	
IMP						0,50	1,25	
RI							0,50	
CF								
SH								

Figure 19: Possible Belbin synergy networks

The simulation specified eight team roles, three target functions, and two relative constraints ($C_x\% \in \{C_t\%, C_c\%\}$).

The relative constraints are calculated by minimal and maximal requirements as follows:

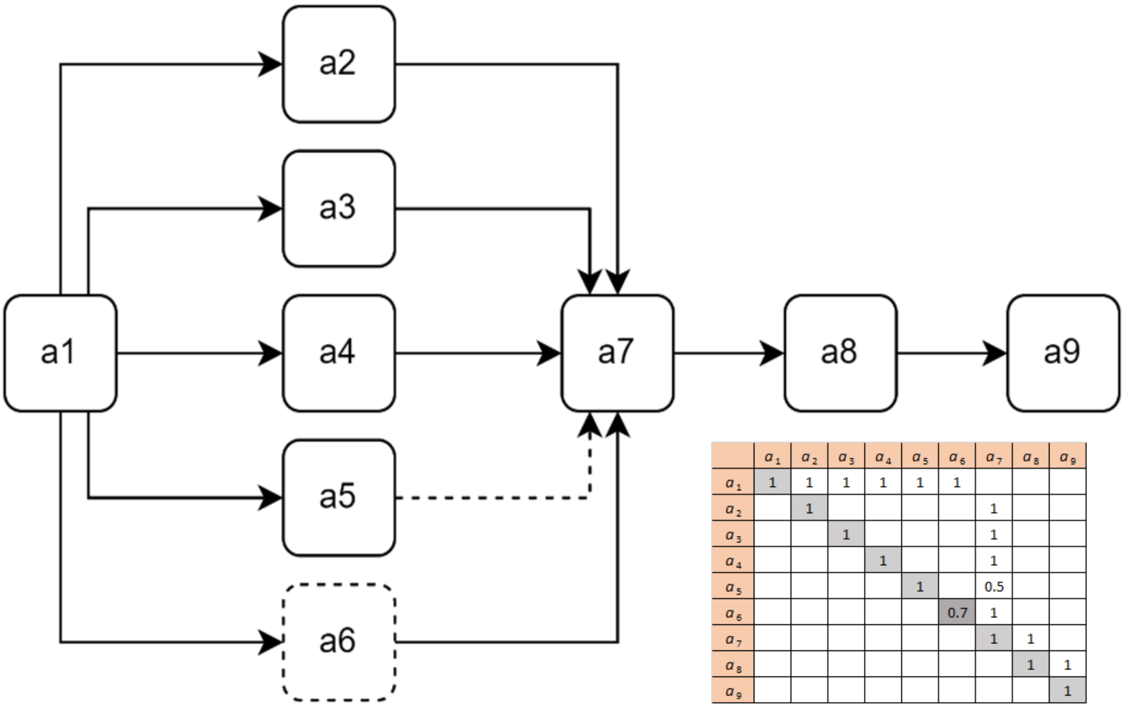
$$C_x\% = \frac{\mathrm{TPX}_{\max} - C_x}{\mathrm{TPX}_{\max} - \mathrm{TPX}_{\min}} \tag{18}$$

where $C_x \in [\mathrm{TPX}_{\max}, \mathrm{TPX}_{\min}]$, $\mathrm{TPX} \in \{\mathrm{TPT}, \mathrm{TPC}\}$. Since, if employees are not assigned to any tasks, the $\mathrm{TPT}_{\max} = \infty$, and $\mathrm{TPC}_{\min} = 0$, the minimal assignment is specified as half of the maximal assignments ($e^{minw} = e^{maxw}/2$). In this way, $\mathrm{TPT}_{\max}$ and $\mathrm{TPC}_{\min}$ can be calculated. In this simulation, $C_x\% \in \{0.5, 0.6, \ldots, 1.0\}$. It provides 6^2 kinds of constraint settings. In every setting, I specified 100 simulations to consider the sensitivity of the estimation of synergy values. Therefore, I obtain $6^2 \cdot 8 \cdot 4 \cdot 100 = 115,200$ SMM matrices.

5.3 Data collection for autonomous team role selection

For the data collection to be able to answer the RQ_3 I used a project from a software development iteration (sprint) consisting of nine tasks, where all but one are mandatory, with the exception being a "comfort feature". Each feature must be implemented and evaluated independently, but in parallel, according to customer requirements. Software integration should begin once all features, except for one that requires a special integration test, have been evaluated.

The examined sprint has nine tasks derived from a project template: design (a_1), implementation of function A (a_2), implementation of function B (a_3), implementation of function C (a_4), implementation of function D (a_5), implementation of extra function E (a_6), testing function E (a_7), maintenance/improvements (a_8), and integration (a_9). This template is used in most sprints. The priorities are determined by DSDM, where mandatory tasks have relative priority, i.e., a task score of 1. Lower priorities have lower task scores. The flexible dependencies came from the technology. The logic network is specified in Figure 20.



	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9
a_1	1	1	1	1	1	1			
a_2		1					1		
a_3			1				1		
a_4				1			1		
a_5					1		0.5		
a_6						0.7	1		
a_7							1	1	
a_8								1	1
a_9									1

Figure 20: Logical structure of task precedences - Autonomous team role selection

Figure 20 shows that there is only one supplementary task (a_6) and one flexible dependency (a_5, a_7). The matching domain, which contains the maximal rates of assignments, was one in each cell.

This set of tasks is typically designed for a team of four and is commonly applied in nearly all software development projects. All team members participated in specialized DISC training, where their behavioral types were identified by a certified trainer. Based on the DISC methodology, four synergy networks were established. The behavioral types of the team members were selected to align with the four DISC behavioral types: dominance (e_1), influence (e_2), steadiness (e_3), and conscientiousness (e_4). The six soft skills assessed were leadership ability (s_1), communication ability (s_2), team player attitude (s_3), problem-solving skills (s_4), interpersonal skills (s_5), and analytical thinking (s_6). The hard skills were the implemented functions per week (s_7), Ux/Ui designs per week (s_8) Ux/Ui, documented functions and test results per week (s_9), and writing deploy and testing scripts per week (s_{10}). Table 21 summarize the required skills from DISC team point of view.

Table 21: Skills versus behavioral types

(S) Skill	Soft skills						Hard skill performances			
Domain	s_1	s_2	s_3	s_4	s_5	s_6	s_7	s_8	s_9	s_{10}
e_1 (D)	1.0	0.8	0.3	0.8	0.8	0.8	4.0	1.5	2.0	1.5
e_2 (I)	0.8	1.0	1.0	0.6	1.0	0.4	0.0	3.0	1.5	1.0
e_3 (S)	0.5	0.5	0.9	0.5	0.3	0.6	2.5	2.0	3.5	3.0
e_4 (C)	0.4	0.5	0.5	1.0	0.4	1.0	5.0	0.0	4.0	1.5

Note. Autonomous team role selection.

Table 22 shows the skill-work requirements for each task.

Table 22: Skilled-work requirements for each task

	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}
a_1	1.00	1.00	1.00	0.35	0.83	0.83	4.00	6.50	9.50	4.00
a_2	0.35	0.38	0.60	0.81	0.63	0.75	10.00	6.00	6.00	4.00
a_3	0.42	0.42	0.80	0.77	0.83	0.83	9.20	3.50	8.00	3.50
a_4	0.38	0.35	0.48	0.96	0.79	0.83	10.50	5.50	5.50	2.50
a_5	0.35	0.38	0.56	0.85	0.63	0.79	9.80	6.00	5.50	3.00
a_6	0.31	0.31	0.40	0.73	0.42	0.88	8.00	6.50	4.00	3.50
a_7	0.81	0.85	0.80	0.38	0.83	1.00	3.50	4.50	2.00	6.00
a_8	0.38	0.77	0.36	1.00	0.42	0.83	10.00	6.00	4.00	4.00
a_9	0.96	0.96	0.40	0.35	1.00	0.17	2.00	1.50	9.50	7.00

Note. Autonomous team role selection.

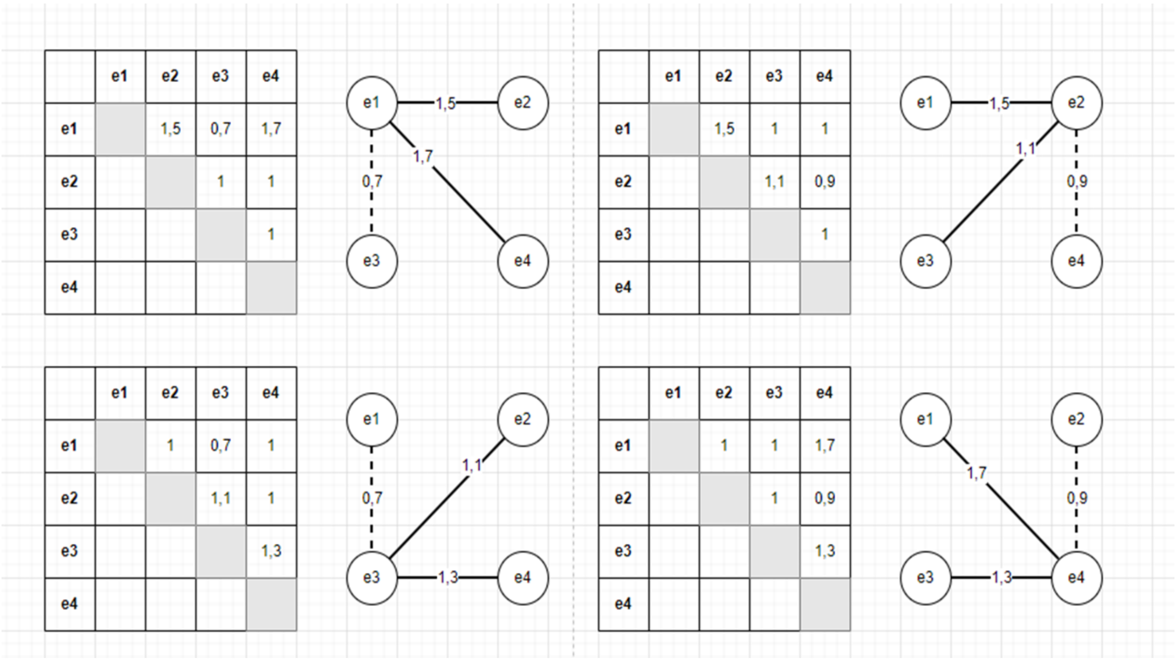
According to Figure 11, based on the former project results and DISC assessment questionnaire by the trainer, the following synergy matrix was specified (see Table 23).

Table 23: Synergy matrix between employees whose behavioral types are categorized by DISC.

	e_1 (D)	e_2 (I)	e_3 (S)	e_4 (C)
e_1 (D)		1.5	0.7	1.7
e_2 (I)			1.1	0.9
e_3 (S)				1.3
e_4 (C)				

Note. Autonomous team role selection. Only upper triangular is specified.

Since the synergy matrix is symmetric and the diagonal values are 1.0, it is sufficient to specify the upper triangular part of the synergy matrix. Table 23 shows that there is positive synergy between employees who have dominance (D) and influence (I) or conscientiousness (C) behavioral types, but there are negative synergies between dominance and steadiness behavioral types. Importantly, to maintain generality, in the simulations, ± 20% are added randomly to every positive and negative synergy value (like in Section 5.2). If a leader for this group is not selected, the team roles are selected autonomously. However, if a leader in a small group is selected, only the synergy of the team leader and the other employees can be dominant; therefore, I obtain four different so-called *dominant synergy networks* (see Figure 21, where only positive or negative synergies are noted).



	e1	e2	e3	e4
e1		1,5	0,7	1,7
e2			1	1
e3				1
e4				

	e1	e2	e3	e4
e1		1,5	1	1
e2			1,1	0,9
e3				1
e4				

	e1	e2	e3	e4
e1		1	0,7	1
e2			1,1	1
e3				1,3
e4				

	e1	e2	e3	e4
e1		1	1	1,7
e2			1	0,9
e3				1,3
e4				

Figure 21: Possible DISC dominant synergy networks

The simulation specified five team roles selection, four target functions, and three relative constraints ($C_x\% \in \{C_s\%, C_t\%, C_c\%\}$).

The relative constraints are calculated by the minimal and maximal requirements as follows:

$$C_x\% = \frac{\mathrm{TPX}_{\max} - C_x}{\mathrm{TPX}_{\max} - \mathrm{TPX}_{\min}} \tag{19}$$

where $C_x \in [\mathrm{TPX}_{\max}, \mathrm{TPX}_{\min}]$, $\mathrm{TPX} \in \{\mathrm{TPT,TPC,TPS}\}$. If employees are not assigned to any of the tasks, $\mathrm{TPT}_{\max} = \infty$, or $\mathrm{TPC}_{\min} = 0$, the minimal assignment is specified as half of the maximal assignments ($e^{minw} = e^{maxw}/2$). In this way, $\mathrm{TPT}_{\max}$ and $\mathrm{TPC}_{\min}$ can be calculated. In this simulation, $C_x\% \in \{0.5, 0.6, \ldots, 1.0\}$. It provides 6^3 kinds of constraint settings. In every setting, I specified 100 simulations to consider the sensitivity of the estimation of synergy values. Therefore, I obtain $6^3 \cdot 5 \cdot 4 \cdot 100 = 432,000$ SMM matrices.

6 Results and Discussion

6.1 Central team roles

Answering to the RQ_2 all 115,200 SMM matrices were analyzed based on the first case. After data cleaning, 115,196 data remained, which do not contain outliers or missing elements. Data loss is less than 10^{-4}. Results of the descriptive statistics can be seen in Table 24.

Table 24: Descriptive analysis

	N	Mean	SD	SE
TPT_{syn}	115, 196	69.632	14.502	0.043
TPT_{nosyn}	115, 196	65.115	12.841	0.038
TPC_{syn}	115, 196	223.096	61.337	0.181
TPC_{nosyn}	115, 196	205.838	50.812	0.150

Table 24 shows that with the consideration of the synergies between the Belbin's team members during the software project scheduling both TPT and TPC can be increased. This shows that it is important to take synergy into account during project scheduling, so that we can estimate the outcome of the project as accurately as possible. This shows an analogy with (Kosztyán et al. 2022) article, where attention was also drawn to the importance of synergy. Using the Anderson-Darling normality test, I demonstrated that the TPT and TPC values do not follow a normal distribution, a finding further confirmed by the Kolmogorov-Smirnov test. Therefore, to assess whether incorporating the synergy network and team roles significantly affects team outcomes, I employed two-tailed non parametric Wilcoxon test for both TPC and TPT.

Table 25: Comparison of TPT and TPC

Measure 1		Measure 2	V	p
TPT_{syn}	-	TPT_{nosyn}	5, 435, 943, 059	< .001
TPC_{syn}	-	TPC_{nosyn}	6, 508, 284, 845	< .001

Note. Wilcoxon test.

Result of the Wilcoxon tests in Table 25 shows that the difference between the TPT_{syn} - TPT_{nosyn} and the TPC_{syn} - TPT_{nosyn} is significant at the (95%) confidence level. So this means that it is important to consider synergistic effects for both TPT and TPC.

After, I examined the relation between the dependent (TPT_{syn}, TPT_{nosyn}, TPC_{syn}, TPC_{nosyn}) and independent (Team selection) variables so that I applied non-parametric, one-tailed Kruskal-Wallis test. Results of the test are shown by Table 26.

Table 26: Results of the Kruskal-Wallis test

(a) Effect of the team selection on the TPC_{syn}

KW - TPC_{syn}	Chi-squared	df	p
Team seletion	97,468	7	$< .001$

(b) Effect of the team selection on the TPT_{syn}

KW - TPT_{syn}	Chi-squared	df	p
Team selection	46,731	7	$< .001$

(c) Effect of the team selection on the TPC_{nosyn}

KW - TPC_{nosyn}	Chi-squared	df	p
Team selection	99,877	7	$< .001$

(d) Effect of the team selection on the TPT_{nosyn}

KW - TPT_{nosyn}	Chi-squared	df	p
Team selection	31,010	7	$< .001$

Based on the results of the Kruskal-Wallis test, the team selection variable is significant for both the project cost and the project duration at the (95%) confidence level. Therefore different teams can be compared with each other based on TPT_{syn}, TPT_{nosyn}), TPC_{syn} and TPC_{nosyn}.

Figure 22 and Figure 23 show the comparison of the groups, arranged in order.

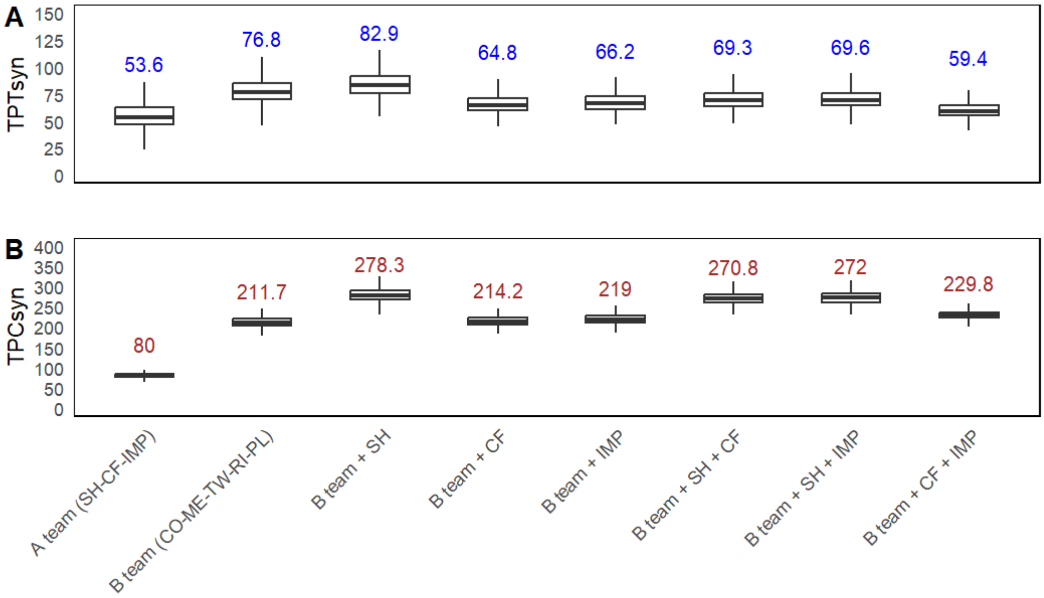


Figure 22: Order of the different team selections considering the synergies. **A** represents the TPT_{syn} and **B** represents the TPC_{syn} which each team achieves during the completion of the project.

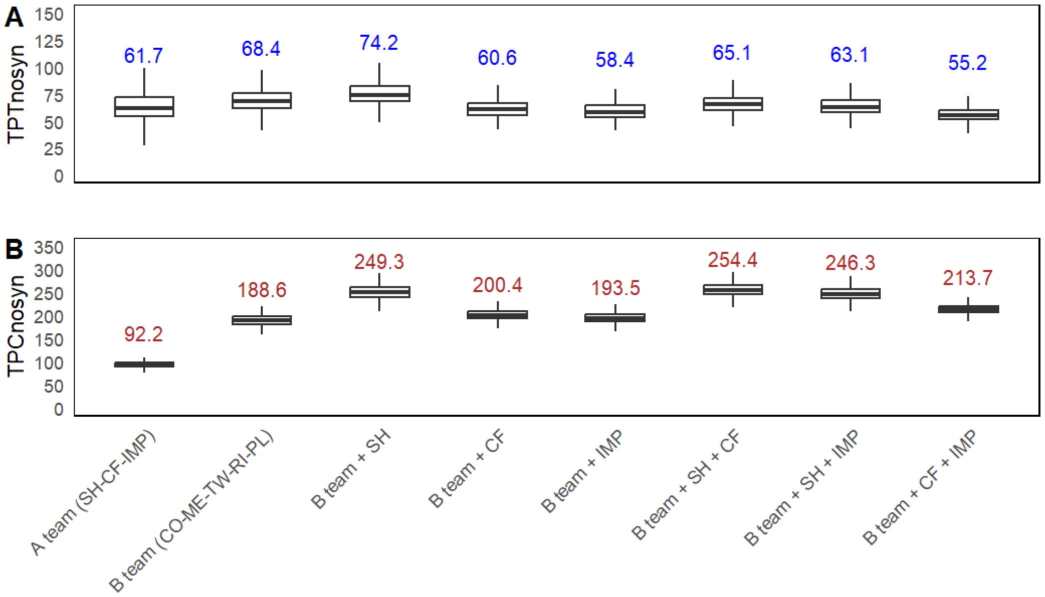


Figure 23: Order of the different team selections neglecting the synergies. **A** represents the TPT$_{nosyn}$ and **B** represents the TPC$_{nosyn}$ which each team achieves during the completion of the project.

As the B part of the Figure 22 and Figure 23 show the lowest TPC with the Action team is earned regardless of whether synergistic effects are considered or not. Besides of that the A parts of these figures show that the Action team achieves the lowest TPT only if synergistic effects are taken into account. If the synergy effects are neglected the Belbin TP + CF + IMP team performs with the lowest TPT. On the other side of the order of the TPC I find the Belbin TP + SH if the synergy effects are considered and the Belbin TP + SH + CF if the synergy effects are neglected. The highest TPT is achieved by the Belbin TP + SH if the synergy effects are considered or neglected.

Individual teams can already be compared based on Figure 22 and Figure 23, but it is important to highlight that the size of the teams differs (the number of members can vary from 3 to 8). Therefore, it is necessary to correct the previous comparison by dividing the results of the teams by the number of their members (normalizing). The TPT per team member is difficult to interpret, so in this case I only examined the TPC per team member (TPCsyn/n or TPCnosyn/n).

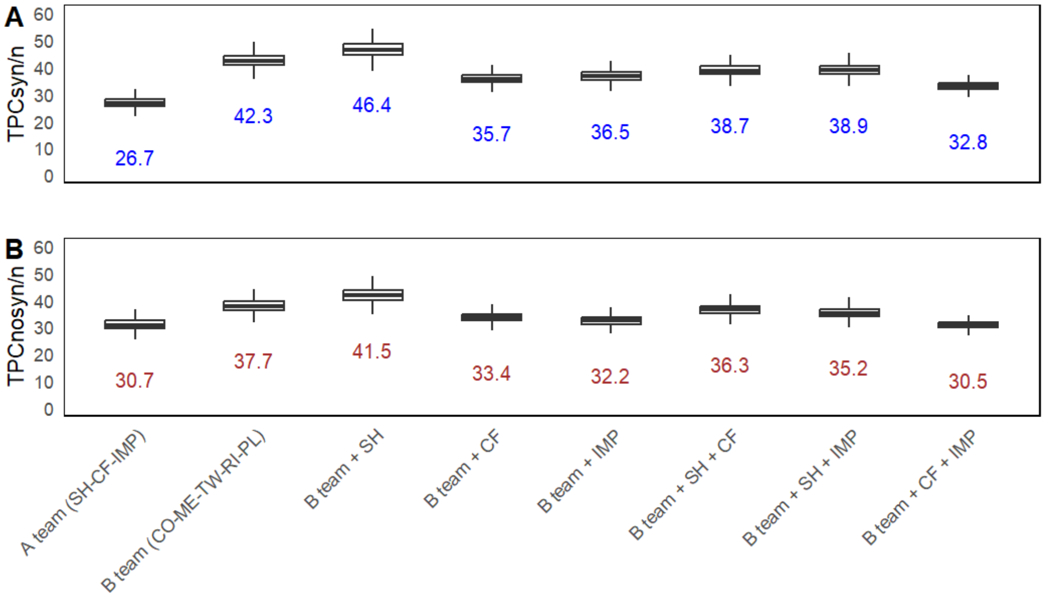


Figure 24: Order of the different team selections regarding the TPC of 1 person and considering (A) (neglecting (B)) the synergies.
Note: n is the number of persons in a group

With the normalized data, Figure 24 shows that the lowest TPC/person can be achived by the Action team if the synergy effects are considered and the Belbin TP + CF + IMP if the synergy effects are neglected. Although the performance of the Action team and the Belbin TP + CF + IMP team is very close even without considering the synergy. Besides that it can be seen that the Belbin TP + SH perform with the highest TPC/person whether the synergistic effects are considered or not and it can also be seen that the ranking of the teams is very similar in both cases.

6.2 Autonomous team role selection

Related to the RQ$_3$, all 432,000 SMM matrices were analyzed. Results of the descriptive statistics can be seen in Table 27(a). Compared to Section 6.1, here the variables follow a normal distribution according to the Kolmogorov-Smirnov test.

Table 27(a)(a) shows the results of the descriptive statistics of project cost (TPC, 1,000 EUR) and duration (TPT, week), and Table 27(a)(b) shows the results of the pairwise t test between considering (syn) and neglecting (nosyn) synergies.

Table 27: Comparison of TPT and TPC

(a) Descriptive statistics

	N	Mean	SD	SE
TPT$_{syn}$	$432,000$	1.947	1.280	1.947
TPT$_{nosyn}$	$432,000$	2.025	1.331	2.025
TPC$_{syn}$	$432,000$	78.932	7.588	1.155
TPC$_{nosyn}$	$432,000$	83.078	8.079	1.232

(b) Paired Samples T-Test

Measure 1		Measure 2	t	df	p
TPT$_{syn}$	-	TPT$_{nosyn}$	-1.000	$431,999$	0.317
TPC$_{syn}$	-	TPC$_{nosyn}$	-52.409	$431,999$	$< .001$

Note. Student's t-test.

Table 27(a)(a) shows that considering positive and negative synergies may reduce both the project duration (TPT) and the project cost TPC. The expected value of TPT is more than two weeks unless synergies are considered. Considering the synergy between employees may reduce the costs to 4,146 EUR. Nevertheless, Table 27(a)(b) about pairwise t-test shows that only the difference in project costs is significant at the (95%) confidence level. Therefore, only TPC is examined.

After, I examined the relation between the dependent (TPC$_{syn}$) and independent (Team role, Target, C_t%, C_c%, C_s%) variables so that I applied ANAlysis Of VAriance (ANOVA). Results of the ANOVA-s are shown by Table28.

Table 28: The effect of team role selection considering the various kinds of target function

ANOVA - TPC$_{syn}$	Sum of Squares	df	Mean Square	F	p
Team role	6,123,182	4	1,534,457	2.655	0.031
Target	1,141,542	3	379,888	0.660	0.577
C_t%	3.254.545	5	649.867	1.129	0.342
C_c%	3,280,458	5	655,003	1.138	0.338
C_s%	2,430,254	5	485,072	0.842	0.519
Residuals	2.49e+11	431,977	575,796		

Table 28 shows that only the team role selection variable is significant for project cost. Neither the target function nor the constraints are significant at the (95%) confidence level.

Figure 25 shows the TPC$_{syn}$ values if the relative cost (C_c%), time (C_t%), and score/scope (C_s%) constraints are neglected.

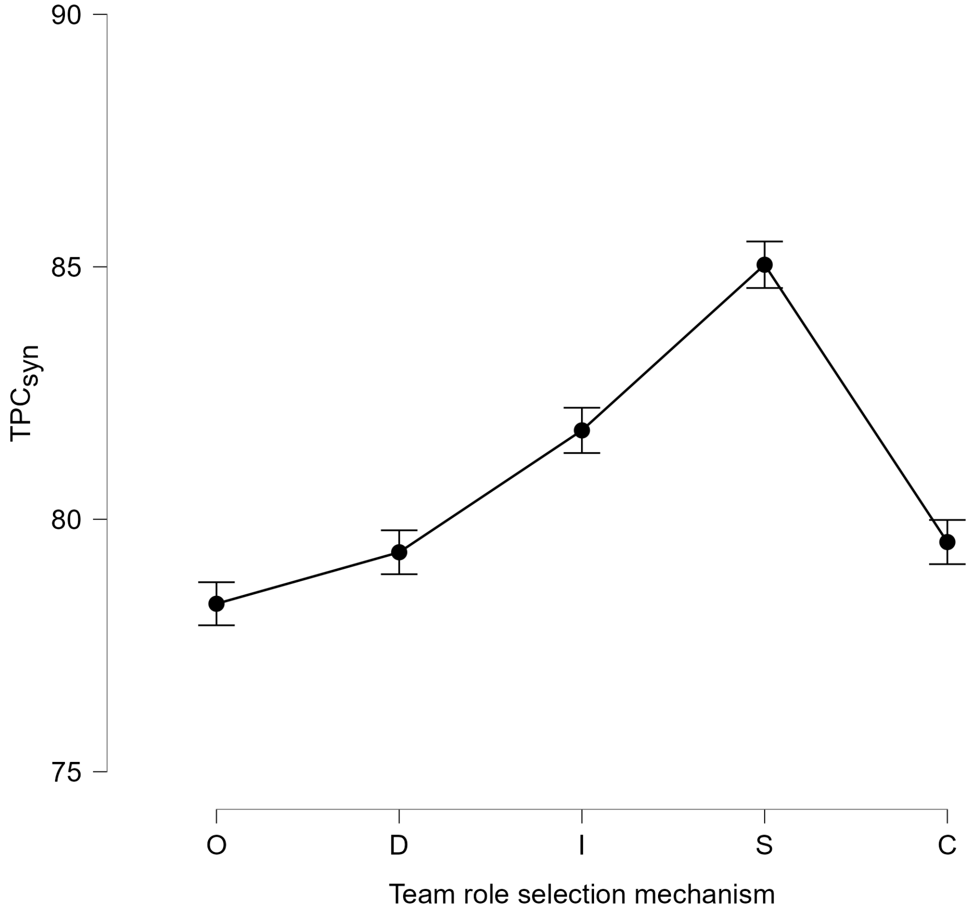


Figure 25: Project costs for various kinds of team role selection mechanisms where there is no constraint ($C_c\% = C_t\% = 1, C_s\% = 0$)

The lowest cost occurs if autonomous (O) team role selection is employed, while the greatest cost occurs if such an employee leads a team whose behavioral type is stable (S).

Although Table 28 shows that the chancing constraints have no significant effect on the project cost, the sensitivity of the role selection to the changes in the constraints was investigated. The Bartlett test showed that changing constraints have no significant effect on the variance of the project cost.

Figure 26 also shows that constraints do not influence the confidence interval of the project cost.

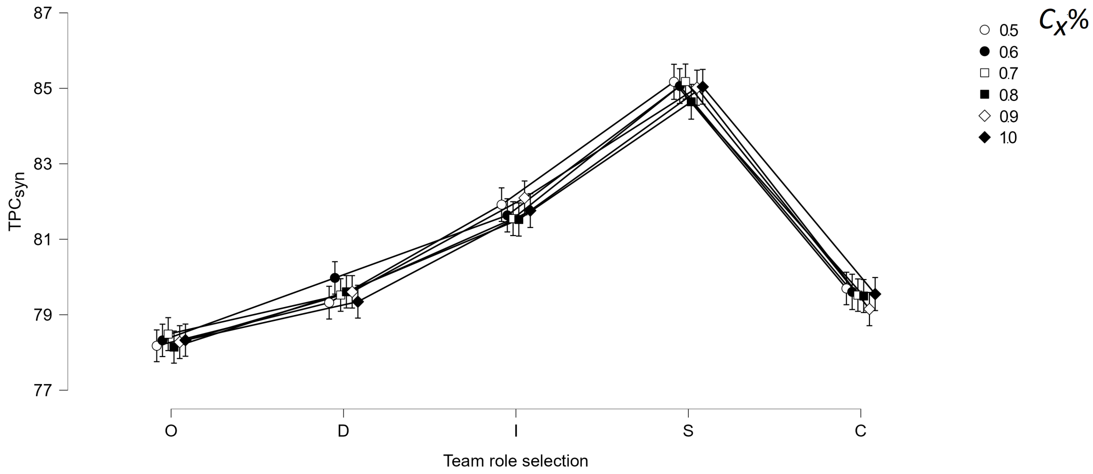


Figure 26: Project costs for various kinds of team role selection mechanisms, where the constraint ($C_c\% = C_t\% = 1 - C_s\%$) is equal

7 Validation and verification

Confirming the answers to the RQ_2 and RQ_3, the simulation results of the case study need to be validated. Validating the simulation results is determining whether the simulation model accurately describes the behavior of the system, in my case the team (Aumann (2007)). The validity of the simulation results should be assessed from both an operational standpoint, ensuring alignment between model output and observed data, and a conceptual perspective, verifying the justifiability of the underlying theory and assumptions. The proposed model uses both stochastic (synergies) and deterministic (others) values which, although I increased the accuracy of the model, the effectiveness of the operational validation is debatable.

In the following subsections, I present the validation process that I used to verify my simulation results. The conceptual validation was done through a literature review, while during the operational validation I used case studies to verify the results for both RQs.

7.1 Conceptual validation

During the conceptual validation, I used the result of as many relevant articles as possible to validate the simulation's result. Since I used the original book of Belbin (Belbin 1981) and DISC (Marston 1928) for the determination of the synergy network between the Belbin's team roles and DISC behavioral types I exclude these papers. Therefore I compared the results of the theses with the result of corresponding papers which were selected by the usage of the PRISMA method (Moher et al. 2009). The corresponding process diagrams are presented in Figure 27. For both cases, I used Scopus as the main database because other databases have less amount of results. The document filtering process was the same in both cases: date-based, keyword-based, document language-based, document type-based, subject area-based, and finally analysis of the remaining papers.

Regarding the papers of Belbin's team roles, when I started the searching with the word "Belbin", I received 15,230 papers. After that I narrowed the search and I received 209 papers for the keyword "Belbin team roles" consists of the title OR the abstract OR one of the keywords. Referring to the fact that the results of older articles were already used, I only examined articles between 2010-2024. I selected only articles or conference papers in English focusing on the following subject areas: Business/Management/Accounting, Computer Science, Engineering, Psychology and Social Science. During the document analysis. Finally 85 documents remained for the document analysis. I first excluded irrelevant studies based on the abstract and then the results sections. Then I found 19 relevant papers where the Belbin's team roles' performances were investigated.

Collecting all relevant papers of DISC behavioral types, I used "DISC personality" keyword for searching them, between 2010-2024, then I received 4.874 results. Similarly in the case of Belbin, I narrowed down the search here as well and I received 39 papers for the keyword "DISC personality" consists of the title OR the abstract OR one of the keywords. I also filtered from these to only English-language articles and conference papers. The selection of subject areas in a similar way to Belbin, was not considered in this case, except that here the leader's style also played a role in the focus of interactions between DISC behavioral types. Finally only 9 paper remains. What is particularly striking about this literature research is that, although DISC behavioral types have been studied by many researcher, investigate the interactions between them is still very incomplete.

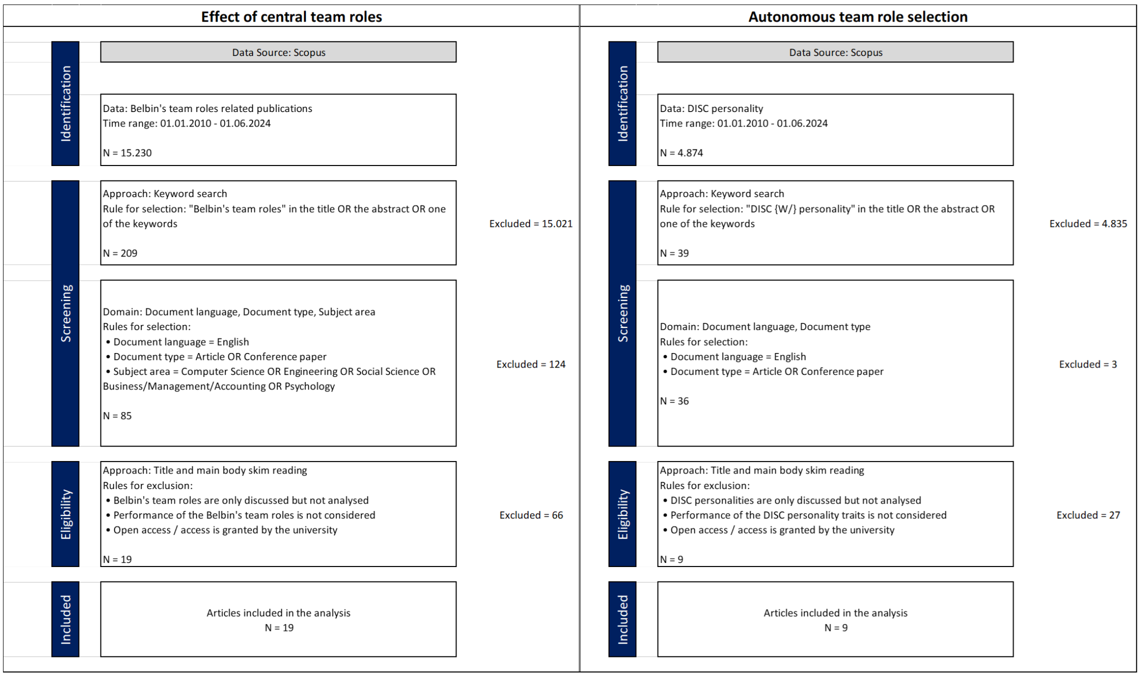

Effect of central team roles
Identification
Data Source: Scopus
Data: Belbin's team roles related publications
Time range: 01.01.2010 - 01.06.2024
N = 15.230
Screening
Approach: Keyword search
Rule for selection: "Belbin's team roles" in the title OR the abstract OR one of the keywords
N = 209
Excluded = 15.021
Domain: Document language, Document type, Subject area
Rules for selection:
• Document language = English
• Document type = Article OR Conference paper
• Subject area = Computer Science OR Engineering OR Social Science OR Business/Management/Accounting OR Psychology
N = 85
Excluded = 124
Eligibility
Approach: Title and main body skim reading
Rules for exclusion:
• Belbin's team roles are only discussed but not analysed
• Performance of the Belbin's team roles is not considered
• Open access / access is granted by the university
N = 19
Excluded = 66
Included
Articles included in the analysis
N = 19
Autonomous team role selection
Identification
Data Source: Scopus
Data: DISC personality
Time range: 01.01.2010 - 01.06.2024
N = 4.874
Screening
Approach: Keyword search
Rule for selection: "DISC {W/} personality" in the title OR the abstract OR one of the keywords
N = 39
Excluded = 4.835
Domain: Document language, Document type
Rules for selection:
• Document language = English
• Document type = Article OR Conference paper
N = 36
Excluded = 3
Eligibility
Approach: Title and main body skim reading
Rules for exclusion:
• DISC personalities are only discussed but not analysed
• Performance of the DISC personality traits is not considered
• Open access / access is granted by the university
N = 9
Excluded = 27
Included
Articles included in the analysis
N = 9

Figure 27: Review method for the selection of the relevant Belbin's team roles and DISC behavioral types papers

7.2 Operational validation

Operational validation was determined as real life validation process for the two participant observation processes: one for validating RQ_2 and another for validating RQ_3. Both validations consist of the same game, the well-known marshmallow challenge game (Wujec (2010)). The main reason for choosing the game was that both individual abilities and characteristics, as well as group and personal behavior, affect the successful completion of the result. The objective of the game is for participants to build, originally in teams of four, the tallest freestanding structure using 20 sticks of spaghetti, one yard of tape, one yard of string, and a marshmallow on top, all within 18 minutes. The team that builds the tallest tower wins. During the game, the available time was reduced to 14 minutes, but before that, for both case studies, a 1-hour familiarization session took place with the participants. Each team received a whole roll of adhesive tape, but only a 50 cm (20 inches) long rope.

In general the key soft skills to win this game are the creativity, the problem-solving, the communication, the collaboration and the leadership. Team members need to communicate to understand each others, they need to collaborate to complete the game within the given time. At least one person on the team should be creative, whose ideas are supported or rejected by at least one problem-solving team member. And last but not least a good leader is very supported to delegates the tasks and brings the team together.

A 4-hours-event was determined and planned for different team games in both cases. Before the Marshmallow challenge game, each committee played a debate game, in which I divided the total population in both cases into 4 teams. The essence of the debate is to discuss a predetermined question, which was as follow:

- Which is better: an electric car or a car with internal combustion engine.

Answering the question is of course not easy, but during the discussion it was easy to find out what kind of behavioral type is in the team and how each person can cooperate. The teams were given 10 minutes to collect their arguments within the teams, and then 20 minutes to discuss them among themselves. The measurement of this argument was not the aim of the dissertation, so I cannot present its results either. The goal was simply for everyone present to get to know each other's behavior and roles in the group. Then I selected the teams according to the simulation teams. In order for the teams to get used to each other as soon as possible, before the Marshmallow challenge game but after the debate I arranged each team separately for a moderated half-hour discussion. The task of each moderator was to promote communication between the members of each team with different games. This section was not taken into account when comparing teams. I did the same in case of the validation of RQ_3, although the invited engineers knew each other from before and had already worked together on several joint tasks.

7.2.1 Central team roles

For validating the results of the RQ_2 I involved 47 person of 120 person from Continental including its software development department but not the same persons I have involved for the simulation. All of them were working as a software development engineer. The first criteria was to involve only motivated people. They were simply asked to join to the validation and if a person agreed this I invited her/him to a planned event. As the second step they filled the Belbin self perception inventory test (validated by Witkowski and Ilski (2000)) to be able to determined their general team role since there were not as much person who was participated in Belbin's training as it was required. Test group was able to perform in the same company as the data for the simulates has been observed. Each person has been identified based on the given IDs from 1 to 47 (1, 2, . . . , 46, 47). Based on the Belbin's test different type of teams were made corresponding to the team compositions (Table 29, Table 30 and Table 31). Groups split into 8 different teams (A, B, C, D, E, F, G, H) corresponding to the 8 different type of teams which I have simulated.

Table 29: Focus groups for validation RQ_2: teams with (Team A) or without (Team B) only action-oriented members

Team ID	Person ID	Dominant Team Role	2nd Dominant Team Role
A	1	SH	IMP
	2	IMP	CF
	3	CF	SH
B	4	CO	TW
	5	PL	TW
	6	RI	TW
	7	ME	CO
	8	TW	RI

Table 30: Focus groups for validation RQ_2: teams with 6 team members

Team ID	Person ID	Dominant Team Role	2nd Dominant Team Role
C	9	CO	IMP
	10	PL	CF
	11	RI	PL
	12	ME	CO
	13	TW	CO
	14	SH	CF
D	15	CO	TW
	16	PL	IMP
	17	RI	PL
	18	ME	SH
	19	TW	IMP
	20	IMP	PL
E	21	CO	PL
	22	PL	CO
	23	RI	IMP
	24	ME	PL
	25	TW	CO
	26	CF	SH

Table 31: Focus groups for validation RQ_2: teams with 7 team members

Team ID	Person ID	Dominant Team Role	2nd Dominant Team Role
F	27	CO	ME
	28	PL	RI
	29	RI	SH
	30	ME	IMP
	31	TW	IMP
	32	SH	CO
	33	IMP	TW
G	34	CO	SH
	35	PL	CF
	36	RI	TW
	37	ME	CF
	38	TW	RI
	39	SH	CO
	40	CF	ME
H	41	CO	ME
	42	PL	SH
	43	RI	TW
	44	ME	CO
	45	TW	RI
	46	IMP	TW
	47	CF	IMP

7.2.2 Autonomous team role selection

For validating the results of the RQ_3 I organized a similar session as in the case of 7.2.1, but involving 20 participants. Each participant was free to choose the session, so that I published a summary of its contents in advance. The one difference between the participants was that, while in the case of the 7.2.1 they already knew each other in advance, here it was specifically required to involve people who had not yet worked together. Compared to the case of Belbin, the second step so that fill in the appropriate questionnaire, which is the questionnaire about DISC behavioral types (DISCtest (2024), validated by Beedu (2021)) was not needed since lots of the software development engineers have participated in DISC training. Each participant received their own identification number, as in the case of Belbin. In this case, however, I used the Roman numerals I-XX.

Based on the DISC test, made by an external company, different type of teams were made corresponding to the team compositions (Table 32). Groups split into 5 different teams (D,I,S,C,O) corresponding to the 5 different type of teams which I have simulated. I asked the group of 20 people to self-organize and form a team of 4 people in such a way that all DISC behavioral types are present, but they share the leadership role in the team (denoted as Group O). Leaders with different DISC behavioral types were selected in the other 4 teams so that each team had all four behavioral types (denoted as Group D,I,S,C accordingly).

Table 32: Focus groups for validation RQ_3

Team ID	Person ID	Dominant Team Role
O	I	D
	II	I
	III	S
	IV	C
D	V	D
	VI	I
	VII	S
	VIII	C
I	IX	D
	X	I
	XI	S
	XII	C
S	XIII	D
	XIV	I
	XV	S
	XVI	C
C	XVII	D
	XVII	I
	XIX	S
	XX	C

8 Results of the validation and discussion

8.1 Central team roles

For the validation of RQ_2 by the literature Table 33 shows the corresponding papers with their contributions to my results and the sample sizes.

Table 33: The examined articles about Belbin's team roles

Reference	Sample Size	Contribution
Abdulrahman et al. (2017)	10 papers were analyzed	In SW engineering the PL and the SH are the most important roles
Aghimien et al. (2018)	47 responders	The preferred team roles are the TW and the CF, the least preferred roles are the PL and the SP
Batenburg et al. (2013)	144 undergraduate master students that participated in 24 teams	No relationship was found between team role diversity and team performance
Boone et al. (2022)	2293 first year master students in medicine	The most preferred role is the TW
Branco et al. (2015)	22 SW engineers	PL and IMP can be good SW project managers
Diab-Bahman (2021)	119 people	The researcher confirmed that an action-oriented role, (neither people and thinking) was not dependent on personality
Dmytro et al. (2017)	Using simulation	CO has a key role but SP can be ignored
Eybers and Hattingh (2019)	101 students	The order of the most prevalent roles 1.: PL; 2.: TW 3.: CF and 4.: CO
Flores Ureba et al. (2022)	149 students enrolled within 21 groups (4-6 member/team)	IMP and CO attained significantly higher scores
García-Ramírez (2020)	Two groups of students participated A (n = 41) and B (n = 25)	CO and SP have a positive relationship with the teams' scores
Isomöttönen and Taipalus (2023)	A total of 21 groups and 79 students participated	SH role is noted for its positive impact on team performance, as teams with a single leader
Liubchenko and Sulimova (2017)	119 persons	The necessary roles for software development teams are those of SH and PL
Monsalves et al. (2023)	A case study of 24 students	During LEGO games IMP and CO are key team roles while SH, PL and CF are unpleasant
Oliver-Quelennec et al. (2022)	31 students	CO intervenes less than the others, the RI is less directive than the others, the TW has more performative speech acts than the CF who did not use that
Omar et al. (2016)	Using simulation	The roles of SH and PL in Belbin Team Roles are significant to the software engineering team; hence the SH role is used for the leader while the PL role is used for other member
Sherstyuk et al. (2016)	Using simulation	CO, ME, IMP and CF have the conservative tendency. PL, RI, SH and TW have the changing tendency
Smith et al. (2017)	145 students	No significant relationship was found between balance Belbin team and team performance, however SH led to increased conflict and, in extreme circumstances
Stankūnas et al. (2012)	55 people	The most preferred team roles are from the action-oriented roles, IMP is the most preferred
Talib et al. (2014)	102 final year students	The most preferred team roles in order are: IMP, CO, SH and TW

According to the literature analysis a software team could be more successful if it contains only leading (in general SH and CO) and working members (in general PL and IMP), if only the performance of the team is considered. Since the synergy take only the performances into account when the synergy effects are considered the most successful software team could be the SH(CO) + PL(IMP) team. However I can conclude that if the quality of the project is also important it is useful to involve further team roles like CF and ME. They can decrease the speed of the implementation and focus on the quality of the product/processes. I still can conclude that if the well-being and conflicts resolution is needed, presence of TW and RI could be mandatory.

Based on the examined studies, I cannot clearly say that in all cases of software projects, action-oriented group has the greatest impact on the success of the software project. Although several studies draw attention to the importance of the presence of SH, CF and IMP in the software team if the goal is to maximize the chance that the project will be completed on time and under a given cost. However, it is easy to see that individual results strongly depend on the size of the examined groups, the examination methods and the examination environment. Furthermore, from the results published in the literature, Belbin's basic theory can be deduced, according to which all Belbin's team roles can have a positive and negative effect on the success of the project. There are places where leaders are more needed and there are places where a good atmosphere is more important, this must always be decided by the company's strategy. In this dissertation, the goal is the success of software projects, in which interactions between team roles are important. With my previously mentioned limitations a summary about my examination can be seen in Table 34.

Table 34: Result of the validation of the central team role's effect on the team performance

Test type	A	B	C	D	E	F	G	H	Constraint
S: Average TPT	55	76	83	68	67	71	70	60	-
S: Rank	1	7	8	4	3	6	5	2	-
M: Time	8:36	14:25	15:32	12:34	12:53	13:50	13:45	11:42	max 14 min
M: Rank	1	7	8	3	4	6	5	2	-
M: Tower height	37 cm	28 cm	44 cm	30 cm	26 cm	38 cm	31 cm	31 cm	min 30 cm

Note. *S*: Simulation, *M*: Marshmallow game
See team compositions in Table 29, Table 30, and Table 31.

Table 34 illustrates that the simulation and case study results align in terms of ranking, with the exception of the Marshmallow Challenge, where Group D and Group E are reversed. In the case studies, I observe that the presence of action-oriented roles significantly impacts team performance in ways consistent with the simulation results.

Unsurprisingly, Team A—the action-oriented team—secured first place in the case studies. This team demonstrated exceptional preparation, clear task understanding, and smooth collaboration, likely due to both positive synergy among members and the advantages of a smaller team size. Furthermore, the team's roles were strongly task-oriented. However, based on team feedback, the atmosphere was less friendly, with members focused solely on the task at hand. Once finished, communication ceased, and team members watched other teams, assessing whether they would surpass their results.

Team H achieved second place. According to team feedback, they reported strong cohesion and a positive atmosphere, with each member assigned specific tasks. However, extended discussions took up a considerable amount of time that could have been spent on work.

Notably, Team C—comprised solely of SH from the action-oriented roles—performed worse than Team B, which lacked any action-oriented roles. This suggests that SH alone may not effectively facilitate group cohesion and mediation as other action-oriented roles might. Literature often attributes this weakness to the fact that SH generally operates alongside CO, allowing for complementary leadership styles. The synergy network further supports that the SH role may be less effective on its own without at least one additional action-oriented role.

8.2 Autonomous team role selection

For the validation of RQ$_3$ by the literature Table 35 shows the corresponding papers with their contributions to my results and the sample sizes.

Table 35: The examined articles about DISC behavioral types

Reference	Sample Size	Contribution
Javahery and Kamali (2023)	20 nonnative EFL teacher	The D and I styles align more closely with leadership roles than the S and C styles. Those with a D style are decisive leaders, individuals with an I style take on the role of a coach.
Shao et al. (2022)	40 residents	No DISC style is "better" than any other, and everyone uses each of the four styles as they go about their daily lives.
Chen et al. (2021)	34 person	No significant difference between the DISC behavioral types but people with I style can escape easily.
Jamjoom et al. (2021)	251 dental students	There is a significant association between behavioral types and GPA, with dominant students often achieving higher grades.
Lykourentzou et al. (2016)	70 people participated, split into 14 groups.	Balanced teams with all DISC behavioral types have more effective communication dynamics and a more motivating environment.
Ahmad et al. (2021)	85 respondents from hospital leaders	There is significant relationship between the DISC behavioral types and leader leadership style with a positive correlation between C style and Laisses-Faire leadership style.
Xia et al. (2017)	346 professionals, in 2 large IT companies	D type is the best for driving a project team with significant difference.
Keogh et al. (2019)	3.396 nurse leaders	During the assessment of the leadership, 73% scored highest in D and C sytles, while the remaining 27% scored highest in preferences for I and S styles.
Slowikowski (2005)	n/a	Individuals with process-oriented behavioral types (D + I styles) excel as leaders during times of change and growth, while those with product-oriented behavioral types (S + C styles) provide essential support and effective leadership in periods of stability.

Based on the literature review summarized in Table 35, it can be concluded that extroversion plays a significant role in the success of projects, suggesting that individuals with D (Dominance) or I (Influence) traits are well-suited to the role of a successful project manager. This likely stems from the fact that individuals with D or I behavioral types tend to possess strong leadership qualities, effective communication skills, and a collaborative, team-oriented attitude (Looi et al. (2011), Orense and Ocampo (2015), Geissler (2014)). Furthermore, while the importance of balanced teams is also reflected within the DISC framework, the primary focus of the literature review was to explore leadership style and self-organization through DISC behavioral types. Therefore, this result serves to reaffirm findings from prior research.

Table 36: Result of the validation of autonomous team selection's effect on team performance

Test type	O	D	I	S	C	Constraint
Simulation: Rank	1	2	4	5	3	-
Marshmallow game: Time	6:32	8:50	12:54	13:45	13:30	max 14 min
Marshmallow game: Rank	1	2	4	5	3	-
Marshmallow game: Tower height	30 cm	40 cm	32 cm	32 cm	36 cm	min 30 cm

See team compositions in Table 32

Table 36 shows that the results of the simulation and the case studies regarding the order are the same. Under this case study the importance of autonomy is also observed and conclude that not only the autonomy but also different type of leader affects significantly the performance of the team therefore the result of the project.

During the Marshmallow challenge game, all teams started to share the responsibilities with the lead of each dedicated leaders, however members of Team O (autonomous team) uses pull-principle and take on the tasks based on their knowledge. Therefore Team O started to build the tower earlier than the other teams. Based on the feedback from the teams the ownership was the highest in case of the Team O because they felt that the product (the tower) is their common goal and all of them are responsible with their skills to build up the tower. In case of the other team (Team D,I,S and C) the first step was to accept the leader and from leader's perspective, define the tasks and responsibilities. I can conclude that it is worth to share the responsibility of leadership among the team members and do not dedicate a leader. Team D reported that the leadership was accepted by the team, however they only concentrated to the accomplishment of the project and the pressure was palpable. While in case of Team I the the focus was on the smooth communication and wasted time because of storytelling. Based on the feedback from Team S the atmosphere was family and and calmly did their work. But in the Team C they were lost in details and overthought the task and did not count with the time constraint.

Based on the simulation results and their validation, I accept the 3nd hypothesis with no limitation.

8.3 Cross-case analysis and results

Since the focus in both cases was on examining the applicability of the same method, it is important to examine the similarities and differences of the results of each case. In both cases the data was provided by the same company. It is noteworthy that the company places an extremely high emphasis on employee education and development. That is why in both cases there was plenty of data to examine them. An important property is that the company's employee capacity goes beyond the multi-skilled workforce and can be said to be multi-personality. Both DISC behavioral types and Belbin's team roles were identified at the company by an external consulting company. In addition, it was possible to determine all the characteristics of each of the employees involved, which was needed for the extended project scheduling method. Thus, in both cases, not only the company, but also the used scheduling method was the same. In both cases, a kind of resource selection process takes place, where the goal is to reveal the differences between teams with different structures and thereby provide feedback for successful team selection.

In both cases, a best team selection process can be demonstrated. While the impact of the action-oriented group of the team formed from Belbin's team roles on the outcome of the project is significant in the examination of central team roles, in the case of a team formed from DISC behavioral types, the selection of different leaders is significant for the success of the project. In both case studies, simulation was primarily used, and then the simulation results were validated. Accordingly, in both cases, teams with different structures can be distinguished in terms of TPC. Although, according to the TPT the individual teams can be distinguished when examining central team roles, but in case of autonomous group selection there is no significant difference between them. In both cases, it was confirmed on the basis of the literature that such research had not been done before, so the results can definitely be said to be new results. In both cases, the simulation results were measured back during the Marshmallow challenge game. According to this, the simulation results were verified in both cases. It can be concluded that with the help of the new method it is possible to test all kinds of personality and behavior types theories. In addition, the method can be a useful project scheduling method for any organization where the personality types or behavioral types or team roles and skill levels of software development professionals are measured.

9 Summary and Conclusion

This dissertation presents and validates a novel, enhanced software project scheduling method. Using this method, two critical aspects of modern software development - namely, the effectiveness of central team roles and autonomous teams - were investigated. Through these efforts, the objectives of the dissertation were achieved, encompassing both the development of the method and the examination of these focal areas. In the following points, I have summarized the fulfillment of individual steps of the dissertation:

[✓] Extended SSPSP was proposed with

[+] flexible dependencies
[+] synergies and behavioral types / team roles of employees
[+] soft skills and hard skills

[✓] Extended SSPSP was validated

[✓] Heterogeneous data from different companies were collected for evaluating the aspect of

[+] Central team roles
[+] Autonomous team role selection

[✓] Both aspects are simulated

[✓] Simulation results of both aspects are validated

9.1 Research theses

Three research theses were formulated in alignment with the research questions, carefully considering both the simulation results (Chapter 6) and their validation (Chapter 8).

RT$_1$ The proposed extended SSPSP method can incorporate Belbin team roles or DISC behavioral types by considering the synergies among these roles, as well as the soft and hard skills they represent, within a flexible or strict software environment. After appropriate hyper-parametrization, the method can also give reliable results on test projects.

RT$_2$ With the usage of the extended SSPSP the importance of the central team roles of the Belbin's team is proved, within the constraints and objective functions defined in the extended SSPSP. Although the presence of IMP and CF roles is certain, the presence of SH is controversial.

RT$_3$ With the usage of the extended SSPSP the positive impact of autonomous teams is proved, within the constraints and objective functions defined in the extended SSPSP.

In summary (Table 37), this research introduces an enhanced model that significantly advances the current understanding of scheduling in software project management. This extended model enables the examination of two previously unexplored aspects through the integration of team dynamics and the refinement of skills into distinct categories. Through the application of the extended SSPSP, team interactions and synergy effects — often overlooked in prior models — can now be incorporated directly into the scheduling framework. Notably, one of the primary limitations of earlier SSPSP versions was their inability to account for synergies among team members. This model addresses this gap by incorporating synergy estimation based on behavioral types and defined team roles, ensuring that both interpersonal and professional dynamics are factored into project timelines.

Furthermore, the model differentiates between hard and soft skills, allowing for a more precise allocation of tasks that align with each team member's unique strengths. This differentiation has led to a refined and more adaptable schedule, as it better reflects the multifaceted skill sets essential in modern software development. Validation results affirm the model's robustness, as the outcomes closely mirror the predictions, underscoring the reliability of both the model structure and the estimated parameters. Therefore, with the validation confirming the model's accuracy and its capacity to make realistic projections, this enhanced SSPSP provides a dependable approach to scheduling that accommodates the complexities of team dynamics and skills in software project environments.

Table 37: Research summary

Item	Statement
RQ1:	How can a software project scheduling method be enhanced to incorporate the uniqueness of different team roles and behavioral types, while also considering their interactions within a heterogeneous network, shaped by diverse skill sets and synergy effects, in both structured and flexible environments?
RA1:	The SSPSP method can be expanded to incorporate Belbin team roles and DISC behavioral types by leveraging the synergies among these roles, as well as the soft and hard skills they represent, within a flexible software environment.
RT1:	The proposed extended SSPSP method can incorporate Belbin team roles or DISC behavioral types by considering the synergies among these roles, as well as the soft and hard skills they represent, within a flexible or strict software environment. After appropriate hyperparameterization, the method can also give reliable results on test projects.
RQ2:	How do central team roles as the central unit of a heterogeneous network influence the success of software projects through their integration into scheduling strategies?
RA2:	The presence of central team roles in software projects positively impacts project success, thereby enhancing performance within the constraints and objective functions defined in the supplemented SSPSP.
RT2:	With the usage of the extended SSPSP the importance of the central team roles of the Belbin's team is proved, within the constraints and objective functions defined in the extended SSPSP. Although the presence of IMP and CF roles is certain, the presence of SH is controversial.
RQ3:	How does autonomously selected team as a heterogeneous network affect the success of software projects through their scheduling?
RA3:	Autonomous teams positively impact the success of software projects, within the constraints and objective functions of the enhanced SSPSP, more effectively than teams with dedicated leaders.
RT3:	With the usage of the extended SSPSP the positive impact of autonomous teams is proved, within the constraints and objective functions defined in the extended SSPSP.

9.2 Contribution to the literature

The dissertation introduced several research findings that offer significant contributions beyond what is currently documented in the literature. Foremost among these is the extension of the SSPSP, leading to the creation of a novel, more refined SPSP approach. While some elements of this new method have been partially addressed individually in past research, no existing model has yet integrated them comprehensively, despite their inherently interconnected nature (Table 38). This newly developed approach, while grounded in established methodologies, presents a unique and cohesive model capable of capturing team dynamics with greater precision and effectively representing the impact of individual contributions on project success. Through this integration, the new SPSP method provides an innovative tool for accurately analyzing the complex interplay of team roles, thereby enhancing the predictive power and applicability of project scheduling strategies.

Table 38: Uniqueness of the extended SSPSP

Source	Multi-skills	Level of skills	Soft skills	Synergy	Personality	Flexibility	Method
Nigar et al. (2023a)	x	-	-	-	-	x	NSGA-II
Stylianou et al. (2012)	-	-	-	-	x	-	NSGA-II
Kosztyán et al. (2022)	x	x	-	x	-	x	Hibrid GA
Li et al. (2024b)	x	x	-	-	x	-	DPRH
Extended SSPSP	x	x	x	x	x	x	Hibrid GA

The establishment of potential synergy networks between DISC behavioral types and Belbin's team roles represents a distinctly innovative contribution. While numerous studies have explored interactions between individual personality types or behavioral types, the concept of synergy networks structured in this way has not yet been articulated in the existing literature. These networks allow for a more nuanced approach to scheduling in software projects, improving precision in task allocation and team configuration. Additionally, synergy networks offer valuable insights into the dynamics between team members, deepening our understanding of how diverse roles and behavioral types interact to influence collective performance. This approach not only enhances project efficiency but also enriches our grasp of interpersonal relationships within teams, providing a powerful framework for future research and practical application in team-based environments.

The two aspects explored in this work, namely the significance of central team roles and the choice of autonomous team roles, are also novel contributions to the literature on software project scheduling. Previous research has largely focused on either optimizing project timelines or maximizing resource utilization, yet the impact of specific team configurations—particularly the influence of core team roles and the presence of autonomous teams—on project outcomes remains largely unexamined. In software project environments, central team roles, such as those with strong influence or critical oversight responsibilities, can shape the trajectory and cohesiveness of the project. However, until now, no studies have rigorously analyzed how these roles specifically contribute to project success, or how their interactions with other roles might drive or hinder performance. Similarly, while autonomous teams - those empowered to make independent decisions and manage their tasks with minimal oversight - are increasingly recognized for their potential benefits, there has been limited research into how the structure of such teams might directly impact the achievement of project goals.

Another significant and novel contribution of this dissertation is the validation of the SPSP method through practical examples—something previously absent in the SPSP literature. Unlike earlier studies, which largely focused on theoretical modeling and simulations without practical testing, this work has validated both the newly proposed method and the results of its applied aspects through real-world examples. This validation process is not only an innovation in its own right but also serves as a strong endorsement of the methodology and findings presented throughout the dissertation.

9.3 Implications and Limitation

The extended SSPSP model, by incorporating multiple factors influencing project team dynamics, offers a strong foundation for further advancements in project scheduling techniques. This model could be enhanced by integrating established concepts such as learning and forgetting curves, or even theories on dynamic team formation, to more accurately reflect real-world team evolution. On the project management side, expanding the model to account for multi-project or project portfolio considerations presents another promising avenue for development. Additionally, the model's accuracy has been thoroughly validated, confirming its capability to accurately capture and analyze diverse project phenomena. This validation not only affirms the method's reliability but also highlights its potential for exploring new phenomena within team-based project settings. The method's applicability has been successfully tested within an industrial context, suggesting it could be highly beneficial for organizations facing human resource challenges, such as high staff turnover. Furthermore, this method has promising implications for enhancing operational efficiency, offering valuable insights and practical solutions for a wide range of industries focused on optimizing human resources and improving project outcomes.

Beyond demonstrating the effectiveness of the proposed method, this dissertation introduces the concept of synergy networks, along with matrix-based data visualization and project scheduling techniques. These contributions enrich the methodological toolkit available for managing and analyzing project teams, especially within the context of software development.

The findings on the two examined aspects also offer substantial practical insights. Further exploration of DISC behavioral types and Belbin's team roles in varied environments could yield even deeper understanding and broader applications. However, the current results are already valuable for software development firms that seek to comprehend why autonomous teams may perform better, at least in the short term, and why identifying and retaining central team members is essential for project success. These insights have the potential to inform team composition strategies, optimize collaboration, and ultimately drive efficiency in team-oriented work environments.

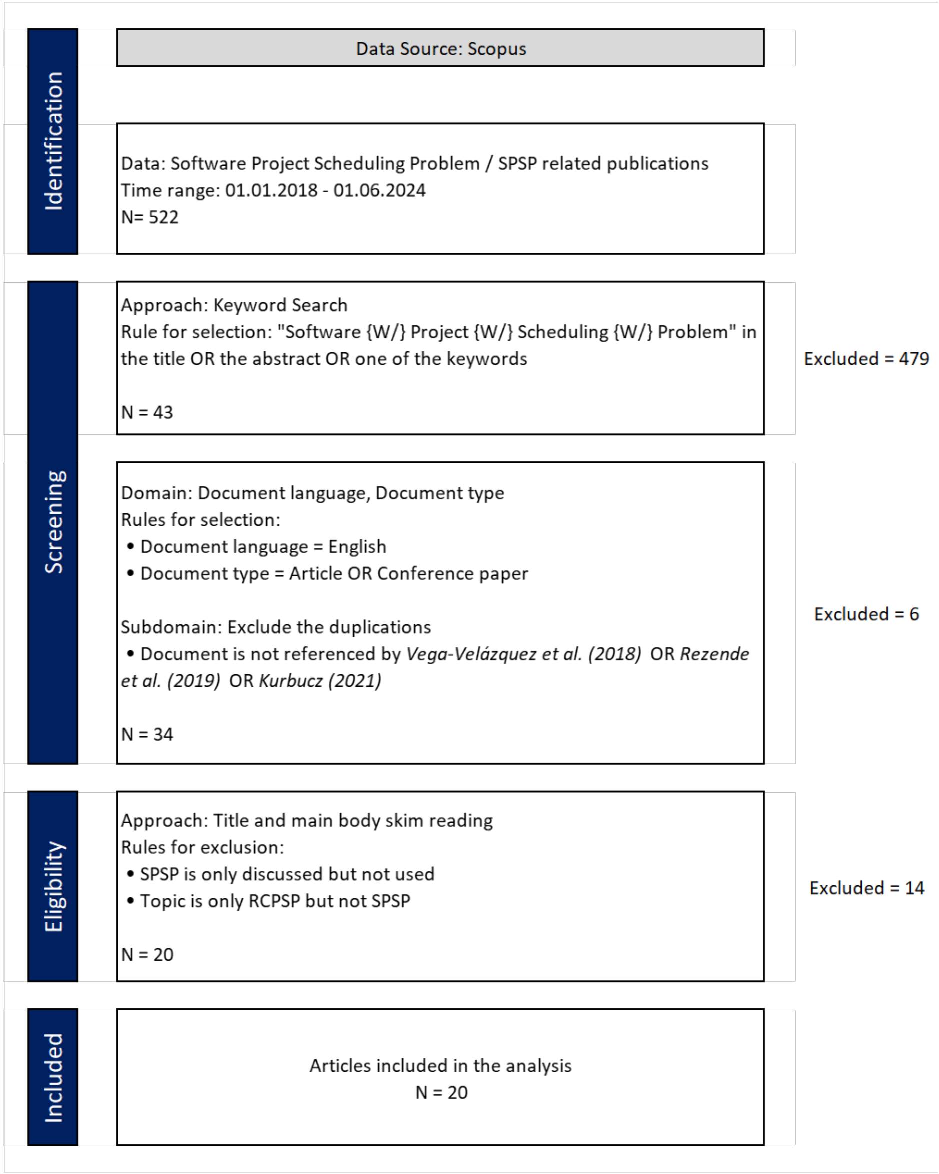
While the method and its specific aspects have been validated, it is important to acknowledge the limitations inherent in the dissertation's findings. The extended model provides accurate results for project scheduling, but it is designed primarily for short-term analysis and does not account for longer-term considerations, such as employee motivation or external economic influences. Consequently, its application is best suited to short-term project planning rather than strategic, long-range projections.

Moreover, while the model builds on advancements in the software project scheduling problem (SPSP), it does not incorporate every factor previously modeled in SPSP literature (see in Table 39). Certain established elements within the scope of SPSP were intentionally excluded in the extended SSPSP to retain focus on specific, immediate project dynamics. Thus, while this model advances current methodologies, it is essential to consider its scope and context when interpreting results and contemplating broader applications.

Appendix A

Systematic Literature Review method on the subject of SPSP

During the first phase I identified 522 papers between 2017 and 2024 which include "Software Project Scheduling Problem" using Scopus as the main database. After the successful identification phase I screened the papers based on different aspects. The first screening rule was the keyword search in the title OR abstract OR the keywords of the papers which resulted in 43 relevant papers. According to the following rule, I filtered out only English-language journal articles and conference articles that were not included in previous reviews (Rezende et al. (2019), Vega-Velázquez et al. (2018), Kurbucz (2021)). Therefore 34 papers remain. After the screening phase and during the eligibility all 34 papers were read to see if they were relevant to SPSP. Finally, 20 relevant papers were included in the literature review in accordance with the PRISMA rules. (see in Picture 28). The conclusion can be drawn from the SPSP articles after 2018 that the main direction is determined by the dynamic environment and stability. The previously modeled learning and forgetting effect often appears, but surprisingly, things like satisfaction, soft skills and uncertainty also appear. I supplemented the Rezende et al. (2019), Vega-Velázquez et al. (2018), Kurbucz (2021) review articles with additional articles from before 2019, which I marked with "d" according to Table 39. Furthermore, Table 39 contains the articles selected using the PRISMA method, marking them with "O" and the articles from Rezende et al. (2019), Vega-Velázquez et al. (2018), Kurbucz (2021) are denoted by "R", "V" ("VR" is used if an article appears in both reviews) and "M" accordingly.



Identification

Data Source: Scopus

Data: Software Project Scheduling Problem / SPSP related publications
Time range: 01.01.2018 - 01.06.2024
N= 522

Screening

Approach: Keyword Search
Rule for selection: "Software {W/} Project {W/} Scheduling {W/} Problem" in the title OR the abstract OR one of the keywords

N = 43

Excluded = 479

Domain: Document language, Document type
Rules for selection:
- Document language = English
- Document type = Article OR Conference paper

Subdomain: Exclude the duplications
- Document is not referenced by *Vega-Velázquez et al. (2018)* OR *Rezende et al. (2019)* OR *Kurbucz (2021)*

N = 34

Excluded = 6

Eligibility

Approach: Title and main body skim reading
Rules for exclusion:
- SPSP is only discussed but not used
- Topic is only RCPSP but not SPSP

N = 20

Excluded = 14

Included

Articles included in the analysis
N = 20

Figure 28: Review method for the selection of the relevant SPSP papers

Table 39: Summary of SPSP literature

Paper	Duration	Cost	Others	Optimization
Alba and Chicano (2005)R	x	x	-	s
Alba and Chicano (2007) VR	x	x	-	s
Almshhadany and Ibrahim (2018) d	x	x	-	m
Alostad (2020) O	x	-	-	s
Alreffaee and Alabajee (2020) M	x	x	-	s

Continued on next page

Table 39 – continued from previous page

Paper	Duration	Cost	Others	Optimization
Alabajee et al. (2021) O	x	x	-	s
Amiri and Barbin (2015) d	x	x	-	m
Antoniol et al. (2004) VR	x	-	-	s
Antoniol et al. (2005) R	x	-	-	s
Bibi et al. (2016) R	x	x	Resource utilization	m
Biju et al. (2015) d	x	x	-	s
Chang et al. (1998) R	x	x	-	s
Chang et al. (2001) VR	x	x	Overtime, Overload	s
Chang et al. (2008) VR	x	x	-	s
Chen and Zhang (2012) (2012) R	-	x	-	s
Cheng et al. (2019) O	x	x	Stability, Robustness, Learning, Forgetting	m
Chicano et al. (2011) VR	x	x	-	m
Chicano et al. (2012) VR	x	x	-	m
Crawford et al. (2014) VR	x	x	-	s
Crawford et al. (2015) d	x	x	-	s
Crawford et al. (2016a) V	x	x	-	s
Crawford et al. (2016b) V	x	x	-	s
Crawford et al. (2018) d	x	x	-	s
Crawford et al. (2019) O	x	x	-	s
de Andrade et al. (2019) M	x	x	-	m
Di Penta et al. (2011) VR	x	-	Fragment	m
Duggan et al. (2004) V	x	-	Quality	m
Dupuy et al. (2013) V	x	x	-	s

Continued on next page

Table 39 – continued from previous page

Paper	Duration	Cost	Others	Optimization
García-Nájera and del Carmen Gómez-Fuentes (2014) R	x	x	Overtime, Overload	m
Ge (2009) R	x	x	Stability	m
Ge and Chang (2006) R	-	x	-	s
Ge and Bin (2016) R	x	x	Stability, Communication	m
Simos et al. (2012) R	x	-	-	s
Gonsalves and Kiyoshi (2010) V	x	x	-	m
Gueorguiev et al. (2009) VR	x	-	Robustness	m
Guo et al. (2019) O	x	x	Learning, Forgetting	m
Hanne and Nickel (2005) V	x	x	Quality	m
Jiang et al. (2007) R	-	x	Risk	m
Jin and Yao (2014) V	x	x	-	s
Kang et al. (2011) R	x	x	-	m
Karthiga and Sumangala (2012) d	x	-	-	s
Kosztyán et al. (2022) O	x	x	Synergy	m
Li et al. (2024b) O	-	x	Uncertainty, Dynamic events	m
Li et al. (2023) O	-	x	Multi-skills	m
Luna et al. (2011) VR	x	x	-	m
Luna et al. (2014) VR	x	x	-	m
Matos and Alba (2012) R	x	x	-	s
Minku et al. (2012) V	x	x	-	s
Minku et al. (2013) VR	x	x	Fragment	s
Myszkowski et al. (2017) R	x	x	-	m
Nigar (2017) d	x	x	Robustness, Uncertainty, Dynamic, Stability, Fragment	m
Nigar et al. (2022) O	x	x	Learning, Employee experience	m

Continued on next page

Table 39 – continued from previous page

Paper	Duration	Cost	Others	Optimization
Nigar et al. (2023a) O	x	x	Overload, Dynamic workforce	m
Nigar et al. (2023b) O	x	x	Dynamic workforce, Employee turnover	m
Ngo-The and Ruhe (2008) V	-	-	Other	s
Ou et al. (2023) O	x	x	People quality	m
Rachman and Ma'sum (2017) d	x	x	-	s
Ren et al. (2011) R	x	-	-	s
Rodríguez et al. (2011) V	x	x	-	m
Sabar et al. (2018) O	x	x	-	m
Shen et al. (2015) VR	x	x	Stability, Robustness	m
Shen et al. (2018) VR	x	x	Stability, Robustness, Satisfaction	m
Shen et al. (2020) O	x	x	Robustness, Employee satisfaction	m
Shen et al. (2024) O	x	x	Stability, Dynamic events, Learning new skills	m
Silva et al. (2020) O	x	x	Robustness, Stability, Dynamic environment	m
Stylianou and Andreou (2013) R	x	-	-	m
Suri and Jajoria (2013) V	x	x	-	s
Szwarc et al. (2023) O	x	x	Multi-skills, Learning and forgetting	m
Szwarc et al. (2024) O	x	x	Uncertainty, Dynamic events, Learning and forgetting	m
Wena and Lin (2008) V	x	x	-	m
Wu et al. (2016) d	x	x	-	m
Wu et al. (2017) VR	x	x	-	m
Xiao et al. (2010) R	x	x	Stability	m
Xiao et al. (2013b) VR	x	x	-	s
Xiao et al. (2015) VR	x	x	-	m
Yannibelli and Amandi (2011) R	-	-	Effectivity level	s

Continued on next page

Table 39 – continued from previous page

Paper	Duration	Cost	Others	Optimization
Yarramsetti and Kousalya (2006) R	x	x	-	m
Zapotecas-Martínez et al. (2020) O	x	x	Agile, Flexibility	m
Zhang et al. (2023) O	x	x	Team satisfaction, Communication, Learning, Different kind of employees, Dynamic workload	m

-s: single objective
-m: multiobjective
-R: references from Rezende et al. (2019)
-V: references from Vega-Velázquez et al. (2018);
-VR: references from both Rezende et al. (2019)
-M: references from Kurbucz (2021)
-d: additionally identified relevant papers before 2019
-O: relevant papers after 2018 using PRISMA method

Bibliography

Abdulrahman, A., Omar, M., Ahmad, M., and Ahmed, V. (2017). An analysis of belbin team roles in software engineering team. *Journal of engineering and Applied Sciences*, 12:6878–6883.

Aghimien, D., Oke, A., Aigbavboa, C., and Ntuli, N. (2018). Preferred team roles of construction professionals in the south african construction industry. In *International Conference on Industrial Engineering and Operations Management Paris, France*, pages 2657–2665.

Ahmad, A., Yuliadi, I., and Pribadi, F. (2021). Disc personality model and leadership style in hospital. *Jurnal Aisyah: Jurnal Ilmu Kesehatan*, 6:261–266.

Akbar, S., Ahmad, I., Khan, R., Lopes, I. O., and Ullah, R. (2022). Multi-skills resource constrained and personality traits based project scheduling. *IEEE Access*, 10:131419–131429.

Alabajee, M. A.-A., Ahmed, D. R., and Alreffaee, T. R. (2021). Solving software project scheduling problem using grey wolf optimization. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, 19(6):1820–1829.

Alba, E. and Chicano, F. (2005). Management of software projects with gas. In *Proceedings of the 6th metaheuristics international conference (MIC'05)*, pages 13–18. Elsevier Science Inc. Vienna, Austria.

Alba, E. and Chicano, J. F. (2007). Software project management with {GAs}. *Information Sciences*, 177(11):2380–2401.

Albusays, K., Bjorn, P., Dabbish, L., Ford, D., Murphy-Hill, E., Serebrenik, A., and Storey, M.-A. (2021). The diversity crisis in software development. *IEEE Software*, 38(2):19–25.

Almshhadany, S. E. and Ibrahim, L. M. (2018). Using multi-objective artificial fish swarm algorithm to solve the software project scheduling problem. *International Journal of Computer Applications*, 181(16):6–13.

Alostad, J. M. (2020). Dynamic software management practices using genetically augmented neural networks. *International Journal of Internet Technology and Secured Transactions*, 10(3):322–336.

Alreffaee, T. R. and Alabajee, M. A.-A. (2020). Solving software project scheduling problem using whale optimization algorithm. In *IOP Conference Series: Materials Science and Engineering*, volume 928/3, page 032084. IOP Publishing.

Amiri, M. and Barbin, J. P. (2015). New approach for solving software project scheduling problem using differential evolution algorithm. *International journal in foundations of computer science & technology (IJFCST)*, 5(1):1–9.

Antill, J. M. and Woodhead, R. W. (1991). *Critical path methods in construction practice*. John Wiley & Sons.

Antoniol, G., Di Penta, M., and Harman, M. (2004). Search-based techniques for optimizing software project resource allocation. In *Genetic and Evolutionary Computation–GECCO 2004: Genetic and Evolutionary Computation Conference, Seattle, WA, USA, June 26-30, 2004. Proceedings, Part II*, pages 1425–1426. Springer.

Antoniol, G., Di Penta, M., and Harman, M. (2005). Search-based techniques applied to optimization of project planning for a massive maintenance project. In *21st IEEE International Conference on Software Maintenance (ICSM'05)*, pages 240–249. IEEE.

Antoniou, A. (2019). Compatibility of small team personalities in computer-based tasks. *Challenges*, 10(1):29.

Aryanee, D., Razali, R., and Mansor, Z. (2020). Team formation for agile software development: a review. *Int. J. Adv. Sci. Eng. Inf. Technol*, 10:2088–5334.

Atkinson, R. (1999). Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria. *International journal of project management*, 17(6):337–342.

Aumann, C. A. (2007). A methodology for developing simulation models of complex systems. *Ecological Modelling*, 202(3-4):385–396.

Baccarini, D. (1999). The logical framework method for defining project success. *Project management journal*, 30(4):25–32.

Balcar, J. (2016). Is it better to invest in hard or soft skills? *The Economic and Labour Relations Review*, 27(4):453–470.

Batenburg, R., van Walbeek, W., et al. (2013). Belbin role diversity and team performance: is there a relationship? *Journal of Management Development*, 32(8):901–913.

Bear, J. B. and Woolley, A. W. (2011). The role of gender in team collaboration and performance. *Interdisciplinary science reviews*, 36(2):146–153.

Beck, K. (1999). Embracing change with extreme programming. *Computer*, 32(10):70–77.

Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., et al. (2001). The agile manifesto.

Beecham, S., Baddoo, N., Hall, T., Robinson, H., and Sharp, H. (2008). Motivation in software engineering: A systematic literature review. *Information and software technology*, 50(9-10):860–878.

Beedu, G. K. (2021). A study on the effectiveness of disc personality test. *Selinus University of Sciences and Literature*.

Belbin, R. M. (1981). *Management Teams: Why They Succeed Or Fail*. Routledge.

Belbin, R. M. (2012). *Team roles at work*. Routledge.

Bell, S. T., Brown, S. G., Colaneri, A., and Outland, N. (2018a). Team composition and the abcs of teamwork. *American psychologist*, 73(4):349.

Bell, S. T., Brown, S. G., and Weiss, J. A. (2018b). A conceptual framework for leveraging team composition decisions to build human capital. *Human Resource Management Review*, 28(4):450–463.

Bell, S. T. and Outland, N. (2017). Team composition over time. In *Team dynamics over time*, pages 3–27. Emerald publishing limited.

Bellenguez, O. and Néron, E. (2004). Lower bounds for the multi-skill project scheduling problem with hierarchical levels of skills. In *International conference on the practice and theory of automated timetabling*, pages 229–243. Springer.

Bellenguez-Morineau, O. and Néron, E. (2007). A branch-and-bound method for solving multi-skill project scheduling problem. *RAIRO-operations Research*, 41(2):155–170.

Bibi, N., Anwar, Z., and Ahsan, A. (2016). Comparison of search-based software engineering algorithms for resource allocation optimization. *Journal of Intelligent Systems*, 25(4):629–642.

Biju, A., Victoire, T. A. A., and Mohanasundaram, K. (2015). [retracted] an improved differential evolution solution for software project scheduling problem. *The Scientific World Journal*, 2015(1):232193.

Binboga, B. and Gumussoy, C. A. (2024). Factors affecting agile software project success. *IEEE Access*.

Binder, J., Aillaud, L. I., and Schilli, L. (2014). The project management cocktail model: An approach for balancing agile and iso 21500. *Procedia-Social and Behavioral Sciences*, 119:182–191.

Blazewicz, J., Lenstra, J. K., and Kan, A. R. (1983). Scheduling subject to resource constraints: classification and complexity. *Discrete applied mathematics*, 5(1):11–24.

Boone, A., Roelants, M., Hoppenbrouwers, K., Vandermeulen, C., Du Bois, M., and Godderis, L. (2022). Perceived team roles of medical students: a five year cross-sectional study. *BMC Medical Education*, 22(1):198.

Borges, G. G. and de Souza, R. C. G. (2024). Skills development for software engineers: Systematic literature review. *Information and Software Technology*, 168:107395.

Boyle, G. J., Matthews, G., and Saklofske, D. H. (2008). Personality theories and models: An overview. *Personality theory and assessment. Personality theories and models*, 1:1–29.

Branco, D. T. M. C., de Oliveira, E. C. C., Galvão, L., Prikladnicki, R., and Conte, T. (2015). An empirical study about the influence of project manager personality in software project effort. In *International Conference on Enterprise Information Systems*, volume 2, pages 102–113. SCITEPRESS.

Bredillet, C. N. (2008). Exploring research in project management: Nine schools of project management research (part 4). *Project management journal*, 39(1):2–6.

Browning, T. R. (2014). Managing complex project process models with a process architecture framework. *International Journal of Project Management*, 32(2):229–241.

Browning, T. R. (2015). Design structure matrix extensions and innovations: a survey and new opportunities. *IEEE Transactions on engineering management*, 63(1):27–52.

Bustamante, A. and Sawhney, R. (2011). Agile xxl: Scaling agile for project teams, seapine software.

Capretz, L. F. and Ahmed, F. (2010). Why do we need personality diversity in software engineering? *ACM SIGSOFT Software Engineering Notes*, 35(2):1–11.

Carruthers, T. (2000). Occupational psychology. *Journal of Occupational and Organizational Psychology*, 73:380.

Chang, C. K., Chao, C., Nguyen, T. T., and Christensen, M. (1998). Software project management net: a new methodology on software management. In *Proceedings. The Twenty-Second Annual*

International Computer Software and Applications Conference (Compsac'98)(Cat. No. 98CB 36241), pages 534–539. IEEE.

Chang, C. K., Christensen, M. J., and Zhang, T. (2001). Genetic algorithms for project management. *Annals of Software Engineering*, 11:107–139.

Chang, C. K., Jiang, H.-y., Di, Y., Zhu, D., and Ge, Y. (2008). Time-line based model for software project scheduling with genetic algorithms. *Information and Software Technology*, 50(11):1142–1154.

Chen, N., Zhao, M., Gao, K., and Zhao, J. (2021). Experimental study on the evaluation and influencing factors on individual's emergency escape capability in subway fire. *International journal of environmental research and public health*, 18(19):10203.

Chen, R., Liang, C., Gu, D., and Leung, J. Y. (2017). A multi-objective model for multi-project scheduling and multi-skilled staff assignment for it product development considering competency evolution. *International Journal of Production Research*, 55(21):6207–6234.

Chen, R., Liang, C., Gu, D., and Zhao, H. (2020). A competence-time-quality scheduling model of multi-skilled staff for it project portfolio. *Computers & Industrial Engineering*, 139:106183.

Chen, W.-N. and Zhang, J. (2012). Ant colony optimization for software project scheduling and staffing with an event-based scheduler. *IEEE Transactions on Software Engineering*, 39(1):1–17.

Cheng, J., Ji, J., Guo, Y.-n., and Ji, J. (2019). Dynamic multiobjective software project scheduling optimization method based on firework algorithm. *Mathematical Problems in Engineering*, 2019(1):8405961.

Chicano, F., Cervantes, A., Luna, F., and Recio, G. (2012). A novel multiobjective formulation of the robust software project scheduling problem. In *Applications of Evolutionary Computation: EvoApplications 2012: EvoCOMNET, EvoCOMPLEX, EvoFIN, EvoGAMES, EvoHOT, EvoIASP, EvoNUM, EvoPAR, EvoRISK, EvoSTIM, and EvoSTOC, Málaga, Spain, April 11-13, 2012, Proceedings*, pages 497–507. Springer.

Chicano, F., Luna, F., Nebro, A. J., and Alba, E. (2011). Using multi-objective metaheuristics to solve the software project scheduling problem. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 1915–1922.

Cleland, D. I. (1994). *Project Management – Strategic Design and Implementation (2nd edn.).* New York, McGraw-Hill.

Coelho, J. and Vanhoucke, M. (2011). Multi-mode resource-constrained project scheduling using rcpsp and sat solvers. *European Journal of Operational Research*, 213(1):73–82.

Conforto, E. C., Salum, F., Amaral, D. C., Da Silva, S. L., and De Almeida, L. F. M. (2014). Can agile project management be adopted by industries other than software development? *Project Management Journal*, 45(3):21–34.

Continental Annual Report (2023). *https://annualreport.continental.com/2023/en/service/docs/annual-report-2023-data.pdf.* Accessed: 17-12-2024.

Cooke-Davies, T. (2002). The "real" success factors on projects. *International journal of project management*, 20(3):185–190.

Corona, E., Pani, F. E., et al. (2013). A review of lean-kanban approaches in the software development. *WSEAS transactions on information science and applications*, 10(1):1–13.

Corsten, H. and Corsten, H. (2000). *Projektmanagement: Einführung*. Oldenbourg Verlag.

Costa, P. and McCrae, R. (1999). A five-factor theory of personality. *Handbook of personality: Theory and research*, 2(01):1999.

Crawford, B., Soto, R., Astorga, G., Castro, C., Paredes, F., Misra, S., and Rubio, J.-M. (2018). Solving the software project scheduling problem using intelligent water drops. *Tehnički vjesnik*, 25(2):350–357.

Crawford, B., Soto, R., Astorga, G., Lemus, J., and Salas-Fernández, A. (2019). Self-configuring intelligent water drops algorithm for software project scheduling problem. In *Information Technology and Systems: Proceedings of ICITS 2019*, pages 274–283. Springer.

Crawford, B., Soto, R., Astorga, G., and Olguín, E. (2016a). An alternative solution to the software project scheduling problem. In *Artificial Intelligence Perspectives in Intelligent Systems: Proceedings of the 5th Computer Science On-line Conference 2016 (CSOC2016), Vol 1*, pages 501–510. Springer.

Crawford, B., Soto, R., Johnson, F., Misra, S., Paredes, F., and Olguín, E. (2015). Software project scheduling using the hyper-cube ant colony optimization algorithm. *Tehnički vjesnik*, 22(5):1171–1178.

Crawford, B., Soto, R., Johnson, F., Monfroy, E., and Paredes, F. (2014). A max–min ant system algorithm to solve the software project scheduling problem. *Expert Systems with Applications*, 41(15):6634–6645.

Crawford, B., Soto, R., Johnson, F., Valencia, C., and Paredes, F. (2016b). Firefly algorithm to solve a project scheduling problem. In *Artificial Intelligence Perspectives in Intelligent Systems: Proceedings of the 5th Computer Science On-line Conference 2016 (CSOC2016), Vol 1*, pages 449–458. Springer.

Cruz, S., Da Silva, F. Q., and Capretz, L. F. (2015). Forty years of research on personality in software engineering: A mapping study. *Computers in Human Behavior*, 46:94–113.

Davis, K. (2017). An empirical investigation into different stakeholder groups perception of project success. *International Journal of Project Management*, 35(4):604–617.

de Andrade, J., Silva, L., Britto, A., and Amaral, R. (2019). Solving the software project scheduling problem with hyper-heuristics. In *Artificial Intelligence and Soft Computing: 18th International Conference, ICAISC 2019, Zakopane, Poland, June 16–20, 2019, Proceedings, Part I 18*, pages 399–411. Springer.

de Azevedo, G. H. I., Pessoa, A. A., and Subramanian, A. (2021). A satisfiability and workload-based exact method for the resource constrained project scheduling problem with generalized precedence constraints. *European Journal of Operational Research*, 289(3):809–824.

De Vreede, T., de Vreede, G.-J., Ashley, G., and Reiter-Palmon, R. (2012). Exploring the effects of personality on collaboration technology transition. In *2012 45th Hawaii International Conference on System Sciences*, pages 869–878. IEEE.

De Wit, A. (1988). Measurement of project success. *International journal of project management*, 6(3):164–170.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197.

Di Penta, M., Harman, M., and Antoniol, G. (2011). The use of search-based optimization techniques to schedule and staff software projects: an approach and an empirical study. *Software: Practice and Experience*, 41(5):495–519.

Diab-Bahman, R. (2021). The impact of dominant personality traits on team roles. *The Open Psychology Journal*, 14(1).

Diekmann, J. and König, C. J. (2018). Personality testing in personnel selection: Love it? leave it? understand it! In *Current issues in work and organizational psychology*, pages 17–31. Routledge.

Dingsøyr, T. and Dybå, T. (2012). Team effectiveness in software development: Human and cooperative aspects in team effectiveness models and priorities for future studies. In *2012 5th international workshop on co-operative and human aspects of software engineering (chase)*, pages 27–29. IEEE.

DISCtest (2024). Disc personality test. validated by Beedu (2021).

Dmytro, L., Katsiaryna, B.-P., Viktor, G., Olexii, K., Andrii, M., and Katerina, D. (2017). Development of the markov model of a project as a system of role communications in a team. *Eastern-European Journal of Enterprise Technologies*, 3(3 (87)):21–28.

Drezet, L.-E. and Billaut, J.-C. (2008). A project scheduling problem with labour constraints and time-dependent activities requirements. *International Journal of Production Economics*, 112(1):217–225.

Duggan, J., Byrne, J., and Lyons, G. (2004). A task allocation optimizer for software construction. *IEEE Software*, 21(3):76–82.

Dupuy, G., Stark, N., and Salto, C. (2013). Algoritmo evolutivo para el problema de planificación en proyectos de desarrollo de software. In *XVIII Congreso Argentino de Ciencias de la Computación*.

Dwertmann, D. J., Nishii, L. H., and Van Knippenberg, D. (2016). Disentangling the fairness & discrimination and synergy perspectives on diversity climate: Moving the field forward. *Journal of Management*, 42(5):1136–1168.

Dybå, T., Dingsøyr, T., and Moe, N. B. (2014). Agile project management. *Software project management in a changing world*, pages 277–300.

Dybå, T. and Dingsøyr, T. (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology*, 50(9):833–859.

Eybers, S. and Hattingh, M. J. (2019). The last straw: Teaching project team dynamics to third-year students. In *ICT Education: 47th Annual Conference of the Southern African Computer Lecturers' Association, SACLA 2018, Gordon's Bay, South Africa, June 18–20, 2018, Revised Selected Papers 47*, pages 237–252. Springer.

Fandel, G., Giese, A., and Mohn, B. (2012). Measuring synergy effects of a public social private partnership (pspp) project. *International Journal of Production Economics*, 140(2):815–824.

Faraj, S. and Sproull, L. (2000). Coordinating expertise in software development teams. *Management science*, 46(12):1554–1568.

Fincher, S., Petre, M., and Clark, M. (2001). *Computer science project work: principles and pragmatics*. Springer Science & Business Media.

Fisher, S. G., Hunter, T., and Macrosson, W. (2001). A validation study of belbin's team roles. *European Journal of Work and Organizational Psychology*, 10(2):121–144.

Flores Ureba, S., Simón de Blas, C., Borrás-Gené, O., and Macías-Guillén, A. (2022). Analyzing the influence of belbin's roles on the quality of collaborative learning for the study of business fundamentals. *Education Sciences*, 12(9):594.

França, A. C. C., Gouveia, T. B., Santos, P. C., Santana, C. A., and da Silva, F. Q. (2011). Motivation in software engineering: A systematic review update. In *15th Annual Conference on Evaluation & Assessment in Software Engineering (EASE 2011)*, pages 154–163. IET.

Galinsky, A. D., Todd, A. R., Homan, A. C., Phillips, K. W., Apfelbaum, E. P., Sasaki, S. J., Richeson, J. A., Olayon, J. B., and Maddux, W. W. (2015). Maximizing the gains and minimizing the pains of diversity: A policy perspective. *Perspectives on Psychological Science*, 10(6):742–748.

García-Nájera, A. and del Carmen Gómez-Fuentes, M. (2014). A multi-objective genetic algorithm for the software project scheduling problem. In Gelbukh, A., Espinoza, F. C., and Galicia-Haro, S. N., editors, *Nature-Inspired Computation and Machine Learning*, pages 13–24, Cham. Springer International Publishing.

García-Ramírez, Y. (2020). Roads project: the relationship between team roles and their performance. In *2020 XV Conferencia Latinoamericana de Tecnologias de Aprendizaje (LACLO)*, pages 1–6. IEEE.

Garousi, V., Tarhan, A., Pfahl, D., Coşkunçay, A., and Demirörs, O. (2019). Correlation of critical success factors with success of software projects: an empirical investigation. *Software Quality Journal*, 27:429–493.

Ge, Y. (2009). Software project rescheduling with genetic algorithms. In *2009 International Conference on Artificial Intelligence and Computational Intelligence*, volume 1, pages 439–443.

Ge, Y. and Bin, X. (2016). Dynamic staffing and rescheduling in software project management: A hybrid approach. *PLoS ONE*, 11(6).

Ge, Y. and Chang, C. (2006). Capability-based project scheduling with genetic algorithms. In *2006 International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06)*, pages 161–161.

Geissler, D. L. (2014). The black and white effect of being labeled: the influence of being labeled by the disc personality model from the perspective of the labeled participants. Master's thesis, University of Twente.

Gilal, A. R., Jaafar, J., Basri, S., Omar, M., and Abro, A. (2016). Impact of software team composition methodology on the personality preferences of malaysian students. In *2016 3rd International Conference on Computer and Information Sciences (ICCOINS)*, pages 454–458. IEEE.

Gonsalves, T. and Kiyoshi, I. (2010). Multi-objective optimization for software development projects. *In Lecture Notes in Engineering and Computer Science: International Multiconference of Engineers and Computer Scientist*, pages 1–6.

Görög, M. (2003). *A projektvezetés mestersége*. Aula, Budapest.

Görög, M. (2007). *Általános projektmenedzsment*. Aula, Budapest.

Görög, M. and Smith, N. J. (1999). Project management for managers. In *PMI Publications*, Sylva, Pennsylvania.

Grossman, R., Nolan, K., Rosch, Z., Mazer, D., and Salas, E. (2022). The team cohesion-performance relationship: A meta-analysis exploring measurement approaches and the changing team landscape. *Organizational Psychology Review*, 12(2):181–238.

Group, T. S. (2015). Chaos report. Technical report, Massachusetts.

Group, T. S. (2021). Chaos report: Beyond infinity. Technical report, Massachusetts.

Groysberg, B., Polzer, J. T., and Elfenbein, H. A. (2011). Too many cooks spoil the broth: How high-status individuals decrease group effectiveness. *Organization Science*, 22(3):722–737.

Gueorguiev, S., Harman, M., and Antoniol, G. (2009). Software project planning for robustness and completion time in the presence of uncertainty using multi objective search based software engineering. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 1673–1680.

Guimera, R., Uzzi, B., Spiro, J., and Amaral, L. A. N. (2005). Team assembly mechanisms determine collaboration network structure and team performance. *Science*, 308(5722):697–702.

Guo, Y., Ji, J., Ji, J., Gong, D., Cheng, J., and Shen, X. (2019). Firework-based software project scheduling method considering the learning and forgetting effect. *Soft computing*, 23:5019–5034.

Gupta, A., Poels, G., and Bera, P. (2022). Using conceptual models in agile software development: a possible solution to requirements engineering challenges in agile projects. *IEEE Access*, 10:119745–119766.

Gutierrez, G., Garzas, J., de Lena, M. T. G., and Moguerza, J. M. (2018). Self-managing: an empirical study of the practice in agile teams. *IEEE software*, 36(1):23–27.

Habibi, F., Barzinpour, F., and Sadjadi, S. (2018). Resource-constrained project scheduling problem: review of past and recent developments. *Journal of project management*, 3(2):55–88.

Hackman, J. R. and Katz, N. (2010). Group behavior and performance. *Handbook of social psychology*, 2:1208–1251.

Hamilton, B. H., Nickerson, J. A., and Owan, H. (2012). Diversity and productivity in production teams. In *Advances in the Economic Analysis of participatory and Labor-managed Firms*, pages 99–138. Emerald Group Publishing Limited.

Hanne, T. and Nickel, S. (2005). A multiobjective evolutionary algorithm for scheduling and inspection planning in software development projects. *European Journal of Operational Research*, 167(3):663–678.

Hanzalek, Z. and Sucha, P. (2017). Time symmetry of resource constrained project scheduling with general temporal constraints and take-give resources. *Annals of Operations Research*, 248(1):209–237.

Hartmann, S. and Briskorn, D. (2022). An updated survey of variants and extensions of the resource-constrained project scheduling problem. *European Journal of operational research*, 297(1):1–14.

Hazır, Ö., Haouari, M., and Erel, E. (2015). Robust optimization for the discrete time-cost tradeoff problem with cost uncertainty. *Handbook on Project Management and Scheduling Vol. 2*, pages 865–874.

Hegazy, T., Shabeeb, A. K., Elbeltagi, E., and Cheema, T. (2000). Algorithm for scheduling with multiskilled constrained resources. *Journal of construction engineering and management*, 126(6):414–421.

Hertel, G. (2011). Synergetic effects in working teams. *Journal of Managerial Psychology*, 26(3):176–184.

Herzberg, F. (1966). Work and the nature of man. *World*.

Herzberg, F. (2008). *One more time: How do you motivate employees?* Harvard Business Review Press.

Hess, A. (2022). Crossing disciplinary borders to improve requirements communication. In *Ernst Denert Award for Software Engineering 2020: Practice Meets Foundations*, pages 115–141. Springer International Publishing Cham.

Hidayati, A., Budiardjo, E. K., and Purwandari, B. (2020). Hard and soft skills for scrum global software development teams. In *Proceedings of the 3rd International Conference on Software Engineering and Information Management*, pages 110–114.

Higgs, M., Plewnia, U., and Ploch, J. (2005). Influence of team composition and task complexity on team performance. *Team Performance Management: An International Journal*, 11(7/8):227–250.

Highsmith, J. and Cockburn, A. (2001). Agile software development: The business of innovation. *Computer*, 34(9):120–127.

Hoda, R., Noble, J., and Marshall, S. (2010). Organizing self-organizing teams. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1*, pages 285–294.

Hoda, R., Noble, J., and Marshall, S. (2012). Self-organizing roles on agile software development teams. *IEEE Transactions on Software Engineering*, 39(3):422–444.

Iriarte, C. and Bayona, S. (2020). It projects success factors: a literature review. *International Journal of Information Systems and Project Management*, 8(2):49–78.

Isomöttönen, V. and Taipalus, T. (2023). Status indicators in software engineering group projects. *Journal of Systems and Software*, 198:111612.

Jamjoom, H., Alshareef, W., and Alotaibi, N. (2021). Association between personality type and academic achievement in undergraduate dental students in king abdulaziz university. *Annals of Dental Specialty*, 9(4-2021):65–72.

Javahery, P. and Kamali, J. (2023). Teachers' personality types and their attitude toward receiving and employing postobservation feedback. *Psychology in the Schools*, 60(8):3073–3089.

Jehn, K. A., Northcraft, G. B., and Neale, M. A. (1999). Why differences make a difference: A field study of diversity, conflict and performance in workgroups. *Administrative science quarterly*, 44(4):741–763.

Jiang, H., Chang, C. K., Xia, J., and Cheng, S. (2007). A history-based automatic scheduling model for personnel risk management. In *31st Annual International Computer Software and Applications Conference (COMPSAC 2007)*, volume 2, pages 361–366. IEEE.

Jin, N. and Yao, X. (2014). Heuristic optimization for software project management with impacts of team efficiency. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 3016–3023. IEEE.

Jugdev, K. and Müller, R. (2005). A retrospective look at our evolving understanding of project success. *Project management journal*, 36(4):19–31.

Kaliprasad, M. (2005). Agile project management: How to succeed in the face of changing project requirements. *Cost Engineering*, 47(10):29.

Kang, D., Jung, J., and Bae, D.-H. (2011). Constraint-based human resource allocation in software projects. *Software: Practice and Experience*, 41(5):551–577.

Kang, H.-R., Yang, H.-D., and Rowley, C. (2006). Factors in team effectiveness: Cognitive and demographic similarities of software development team members. *Human Relations*, 59(12):1681–1710.

Kanski, L., Budzynska, K., and Chadam, J. (2023). The impact of identified agility components on project success—ict industry perspective. *Plos one*, 18(3):e0281936.

Karthiga, V. and Sumangala, K. (2012). A hybrid approach for software project scheduling. *International Journal of Computer Applications*, 59(16).

Katzenbach, J. R. and Smith, D. K. (2015). *The wisdom of teams: Creating the high-performance organization*. Harvard Business Review Press.

Kazemifard, M., Zaeri, A., Ghasem-Aghaee, N., Nematbakhsh, M. A., and Mardukhi, F. (2011). Fuzzy emotional cocomo ii software cost estimation (fecsce) using multi-agent systems. *Applied Soft Computing*, 11(2):2260–2270.

Kellenbrink, C. and Helber, S. (2015). Scheduling resource-constrained projects with a flexible project structure. *European Journal of Operational Research*, 246(2):379–391.

Kelley Jr, J. E. (1961). Critical-path planning and scheduling: Mathematical basis. *Operations research*, 9(3):296–320.

Keogh, T. J., Robinson, J. C., and Parnell, J. M. (2019). Assessing behavioral styles among nurse managers: Implications for leading effective teams. *Hospital topics*, 97(1):32–38.

Kia, R., Shahnazari-Shahrezaei, P., and Zabihi, S. (2016). Solving a multi-objective mathematical model for a multi-skilled project scheduling problem by cplex solver. In *2016 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 1220–1224. IEEE.

Kniberg, H. and Skarin, M. (2010). *Kanban and Scrum-making the most of both*. Lulu. com.

Kolisch, R. and Heimerl, C. (2012). An efficient metaheuristic for integrated scheduling and staffing it projects based on a generalized minimum cost flow network. *Naval Research Logistics (NRL)*, 59(2):111–127.

Koops, L., Bosch-Rekveldt, M., Coman, L., Hertogh, M., and Bakker, H. (2016). Identifying perspectives of public project managers on project success: Comparing viewpoints of managers from five countries in north-west europe. *International journal of project management*, 34(5):874–889.

Kosztyán, Z. T. (2012). Challenges of the project planning methods in the 21st century. *Problems of Management in the 21st Century*, 5(1):46–60.

Kosztyán, Z. T. (2016). Projektek és üzleti folyamatok tervezése, nyomonkövetése.

Kosztyán, Z. T. (2022). Mfpp: Matrix-based flexible project planning. *SoftwareX*, 17:100973.

Kosztyán, Z. T., Bogdány, E., Szalkai, I., and Kurbucz, M. T. (2022). Impacts of synergies on software project scheduling. *Annals of Operations Research*, 312:883–908.

Kosztyán, Z. T. and Szalkai, I. (2020). Multimode resource-constrained project scheduling in flexible projects. *Journal of Global Optimization*, 76(1):211–241.

Kreter, S., Rieck, J., and Zimmermann, J. (2016). Models and solution procedures for the resource-constrained project scheduling problem with general temporal constraints and calendars. *European Journal of Operational Research*, 251(2):387–403.

Kurbucz, M. T. (2021). *Synergy-based software project scheduling problem: formalization, simulation, and solution*. PhD thesis, Pannon Egyetem.

Larson Jr, J. R. (2013). *In search of synergy in small group performance*. Psychology Press.

Lee, D., Huh, Y., and Reigeluth, C. M. (2015). Collaboration, intragroup conflict, and social skills in project-based learning. *Instructional science*, 43:561–590.

Lewin, K. (1947). Frontiers in group dynamics: Concept, method and reality in social science; social equilibria and social change. *Human relations*, 1(1):5–41.

Leyman, P., Van Driessche, N., Vanhoucke, M., and De Causmaecker, P. (2019). The impact of solution representations on heuristic net present value optimization in discrete time/cost trade-off project scheduling with multiple cash flow and payment models. *Computers & Operations Research*, 103:184–197.

Li, C., Wang, F., and Chung, T. (2024a). Multi-mode multi-skill resource-constrained project scheduling problem with differentiated professional capabilities. *Journal of Project Management*, 9(1):27–44.

Li, H. and Zhang, H. (2013). Ant colony optimization-based multi-mode scheduling under renewable and nonrenewable resource constraints. *Automation in construction*, 35:431–438.

Li, H., Zhu, H., Zheng, L., and Liu, Y. (2023). Software project scheduling with multitasking. *Economic Computation & Economic Cybernetics Studies & Research*, 57(1).

Li, H., Zhu, H., Zheng, L., and Xie, F. (2024b). Software project scheduling under activity duration uncertainty. *Annals of Operations Research*, 338(1):477–512.

Liemhetcharat, S. and Veloso, M. (2012). Modeling and learning synergy for team formation with heterogeneous agents. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 365–374.

Liemhetcharat, S. and Veloso, M. (2014). Weighted synergy graphs for effective team formation with heterogeneous ad hoc agents. *Artificial Intelligence*, 208:41–65.

Liubchenko, V. and Sulimova, I. (2017). Examining the attributes of transitions between team roles in the software development projects. *Eastern-European Journal of Enterprise Technologies*, 3(1):12–17.

Lix, K., Goldberg, A., Srivastava, S. B., and Valentine, M. A. (2022). Aligning differences: Discursive diversity and team performance. *Management Science*, 68(11):8430–8448.

Looi, Q. E., See, S. L., Tay, C. S., and Ng, G. K. (2011). Understanding users by their disc personality through interactive gaming. In *HCI International 2011–Posters' Extended Abstracts: International Conference, HCI International 2011, Orlando, FL, USA, July 9-14, 2011, Proceedings, Part I 14*, pages 312–316. Springer.

Luna, F., González-Álvarez, D. L., Chicano, F., and Vega-Rodríguez, M. A. (2011). On the scalability of multi-objective metaheuristics for the software scheduling problem. In *2011 11th International Conference on Intelligent Systems Design and Applications*, pages 1110–1115. IEEE.

Luna, F., González-Álvarez, D. L., Chicano, F., and Vega-Rodríguez, M. A. (2014). The software project scheduling problem: A scalability analysis of multi-objective metaheuristics. *Applied Soft Computing*, 15:136–148.

Lundin, R. A. and Söderholm, A. (1995). A theory of the temporary organization. *Scandinavian Journal of management*, 11(4):437–455.

Lykourentzou, I., Antoniou, A., Naudet, Y., and Dow, S. P. (2016). Personality matters: Balancing for personality types leads to better outcomes for crowd teams. In *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*, pages 260–273.

Maghsoudlou, H., Afshar-Nadjafi, B., and Niaki, S. T. A. (2016). A multi-objective invasive weeds optimization algorithm for solving multi-skill multi-mode resource constrained project scheduling problem. *Computers & chemical engineering*, 88:157–169.

Maghsoudlou, H., Afshar-Nadjafi, B., and Niaki, S. T. A. (2017). Multi-skilled project scheduling with level-dependent rework risk; three multi-objective mechanisms based on cuckoo search. *Applied Soft Computing*, 54:46–61.

Malcolm, D. G., Roseboom, J. H., Clark, C. E., and Fazar, W. (1959). Application of a technique for research and development program evaluation. *Operations research*, 7(5):646–669.

Marston, W. M. (1928). Review of a short outline of comparative psychology. *Journal of Abnormal and Social Psychology*, 23(2):256–257.

Maslow, A. (1943). A theory of human motivation. *Psychological Review google schola*, 2:21–28.

Matos, J. and Alba, E. (2012). Benchmarking chc on a new application: the software project scheduling problem. In *Parallel Problem Solving from Nature-PPSN XII: 12th International Conference, Taormina, Italy, September 1-5, 2012, Proceedings, Part II 12*, pages 448–457. Springer.

Matthies, C., Huegle, J., Dürschmid, T., and Teusner, R. (2019). Attitudes, beliefs, and development data concerning agile software development practices. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, pages 158–169. IEEE.

Matturro, G. (2013). Soft skills in software engineering: A study of its demand by software companies in uruguay. In *2013 6th international workshop on cooperative and human aspects of software engineering (CHASE)*, pages 133–136. IEEE.

Matturro, G., Raschetti, F., and Fontán, C. (2019). A systematic mapping study on soft skills in software engineering. *J. Univers. Comput. Sci.*, 25(1):16–41.

Meckenstock, J.-N. (2024). Shedding light on the dark side–a systematic literature review of the issues in agile software development methodology use. *Journal of Systems and Software*, page 111966.

Mehta, N., Hall, D., and Byrd, T. (2014). Information technology and knowledge in software development teams: The role of project uncertainty. *Information & Management*, 51(4):417–429.

Meier, A. and Kock, A. (2023). The human factor in agility: Exploring employee dedication in agile project organizations. *International Journal of Project Management*, 41(7):102527.

Melo, C. d. O., Santana, C., and Kon, F. (2012). Developers motivation in agile teams. In *2012 38th Euromicro Conference on Software Engineering and Advanced Applications*, pages 376–383. IEEE.

Meslec, N. and Curşeu, P. L. (2015). Are balanced groups better? belbin roles in collaborative learning groups. *Learning and Individual Differences*, 39:81–88.

Minku, L. L., Sudholt, D., and Yao, X. (2012). Evolutionary algorithms for the project scheduling problem: runtime analysis and improved design. In *Proceedings of the 14th annual conference on Genetic and evolutionary computation*, pages 1221–1228.

Minku, L. L., Sudholt, D., and Yao, X. (2013). Improved evolutionary algorithm design for the project scheduling problem based on runtime analysis. *IEEE Transactions on Software Engineering*, 40(1):83–102.

Misra, S. C., Kumar, V., and Kumar, U. (2006). Success factors of agile software development. *Software engineering research and practice*, 1:233–239.

Moher, D., Liberati, A., Tetzlaff, J., Altman, D. G., and PRISMA Group*, t. (2009). Preferred reporting items for systematic reviews and meta-analyses: the prisma statement. *Annals of internal medicine*, 151(4):264–269.

Monsalves, D., Cornide-Reyes, H., and Riquelme, F. (2023). Relationships between social interactions and belbin role types in collaborative agile teams. *IEEE Access*, 11:17002–17020.

Mostert, N. M. (2015). Belbin-the way forward for innovation teams. *Journal of Creativity & Business Innovation*, 1.

Mtsweni, E. S., Hörne, T., and van der Poll, J. A. (2016). Soft skills for software project team members. *International Journal of Computer Theory and Engineering*, 8(2):150.

Mumford, T. V. and Mattson, M. (2009). Will teams work?: how the nature of work drives synergy in autonomous team designs. In *Academy of Management Proceedings*, volume 2009, pages 1–6. Academy of Management Briarcliff Manor, NY 10510.

Muniz, M. and Flamand, T. (2023). Sports analytics for balanced team-building decisions. *Journal of the Operational Research Society*, 74(8):1892–1909.

Myers, I. B. (1962). *The Myers-Briggs Type Indicator: Manual (1962).* Consulting Psychologists Press.

Myszkowski, P. B., Laszczyk, M., and Lichodij, J. (2017). Efficient selection operators in nsga-ii for solving bi-objective multi-skill resource-constrained project scheduling problem. In *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 83–86. IEEE.

Myszkowski, P. B., Laszczyk, M., Nikulin, I., and Skowroński, M. (2019). imopse: a library for bicriteria optimization in multi-skill resource-constrained project scheduling problem. *Soft Computing*, 23:3397–3410.

Myszkowski, P. B., Skowroński, M. E., Olech, Ł. P., and Oślizło, K. (2015a). Hybrid ant colony optimization in solving multi-skill resource-constrained project scheduling problem. *Soft Computing*, 19:3599–3619.

Myszkowski, P. B., Skowroński, M. E., and Podlodowski, Ł. (2013). Novel heuristic solutions for multi-skill resource-constrained project scheduling problem. In *2013 federated conference on computer science and information systems*, pages 159–166. IEEE.

Myszkowski, P. B., Skowroński, M. E., and Sikora, K. (2015b). A new benchmark dataset for multi-skill resource-constrained project scheduling problem. In *2015 federated conference on computer science and information systems (FedCSIS)*, pages 129–138. IEEE.

Napier, N. P., Keil, M., and Tan, F. B. (2009). It project managers' construction of successful project management practice: a repertory grid investigation. *Information Systems Journal*, 19(3):255–282.

Ngo-The, A. and Ruhe, G. (2008). Optimized resource allocation for software release planning. *IEEE Transactions on Software Engineering*, 35(1):109–123.

Nigar, N. (2017). Model-based dynamic software project scheduling. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pages 1042–1045.

Nigar, N., Shahzad, M. K., Islam, S., Kumar, S., and Jaleel, A. (2022). Modeling human resource experience evolution for multiobjective project scheduling in large scale software projects. *IEEE Access*, 10:44677–44690.

Nigar, N., Shahzad, M. K., Islam, S., Oki, O., and Lukose, J. M. (2023a). Multi-objective dynamic software project scheduling: A novel approach to handle employee's addition. *IEEE Access*, 11:39792–39806.

Nigar, N., Shahzad, M. K., Islam, S., Oki, O., and Lukose, J. M. (2023b). A novel multi-objective evolutionary algorithm to address turnover in the software project scheduling problem based on best fit skills criterion. *IEEE Access*, 11:89742–89756.

Norhanim, Z. et al. (2019). The impact of virtual team characteristics on project effectiveness. *Journal of Design+ Built*, 12(1).

Ojha, A. K. (2005). Impact of team demography on knowledge sharing in software project teams. *south asian journal of Management*, 12(3):67.

Oliver-Quelennec, K., Bouchet, F., Carron, T., and Pinçon, C. (2022). Understanding online collaboration through speech acts associated to belbin profiles. In *CSCL 2022-15th International Conference on Computer-Supported Collaborative Learning*, pages 324–327.

Olsen, R. P. (1971). Can project management be defined? *Project Management Quarterly*, 2(1):12–14.

Omar, M., Ahmad Khasasi, N. L., Syed Abdullah, S. L., Hashim, N. L., Romli, R., and Katuk, N. (2018). Defining skill sets requirements for agile scrum team formation. *Journal of Engineering and Applied Sciences*, 13(3):784–789.

Omar, M., Hasan, B., Ahmad, M., Yasin, A., Baharom, F., Mohd, H., and Darus, N. M. (2016). Towards a balanced software team formation based on belbin team role using fuzzy technique. In *AIP Conference Proceedings*, volume 1761. AIP Publishing.

Orense, R. and Ocampo, R. (2015). Correlation on the disc personality profile and leadership styles of the student leaders of cas in ay 2014-2015. *The Bedan Journal of Psychology*, 1:90–101.

Ou, C.-H., Li, Y.-H., Chen, C.-Y., Wu, C.-H., Tsai, Y.-C., Yan, Z.-Y., and Chang, C.-R. (2023). Quantum-inspired optimization for task scheduling in software development projects. In *2023 IEEE International Conference on Quantum Computing and Engineering (QCE)*, volume 2, pages 348–349. IEEE.

Pankratz, O. and Basten, D. (2018). Opening the black box: Managers' perceptions of is project success mechanisms. *Information & Management*, 55(3):381–395.

Pant, I. and Baroudi, B. (2008). Project management education: The human skills imperative. *International journal of project management*, 26(2):124–128.

Persson, J. and Mathiassen, L. (2009). A process for managing risks in distributed teams. *IEEE software*, 27(1):20–29.

Peslak, A. R. (2006). The impact of personality on information technology team projects. In *Proceedings of the 2006 ACM SIGMIS CPR conference on computer personnel research: Forty four years of computer personnel research: achievements, challenges & the future*, pages 273–279.

Phillips, J. (2018). *PMP Project Management Professional Study Guide, Fifth Edition*. McGraw Hill.

Phillips, K. W., Liljenquist, K. A., and Neale, M. A. (2009). Is the pain worth the gain? the advantages and liabilities of agreeing with socially distinct newcomers. *Personality and Social Psychology Bulletin*, 35(3):336–350.

Pich, M. T., Loch, C. H., and Meyer, A. D. (2002). On uncertainty, ambiguity, and complexity in project management. *Management science*, 48(8):1008–1023.

Pieterse, V., Leeu, M., and van Eekelen, M. (2018). How personality diversity influences team performance in student software engineering teams. In *2018 Conference on Information Communications Technology and Society (ICTAS)*, pages 1–6. IEEE.

Pinto, J. K. and Slevin, D. P. (1988). Critical success factors across the project life cycle. *Project Management Journal*, 19(3):67–75.

Plattner, H., Meinel, C., and Weinberg, U. (2009). *Design thinking*. Springer.

PMI (2017). *A guide to the project management body of knowledge: Sixth edition*. Project Management Institute, Inc.

Pritsker, A. A. B., Waiters, L. J., and Wolfe, P. M. (1969). Multiproject scheduling with limited resources: A zero-one programming approach. *Management science*, 16(1):93–108.

Rachman, V. and Ma'sum, M. A. (2017). Comparative analysis of ant colony extended and mix-min ant system in software project scheduling problem. In *2017 International Workshop on Big Data and Information Security (IWBIS)*, pages 85–91. IEEE.

Rajendran, M. (2005). Analysis of team effectiveness in software development teams working on hardware and software environments using belbin self-perception inventory. *Journal of Management Development*, 24(8):738–753.

Reeves, C. R. and Wright, C. C. (1999). *Genetic Algorithms and the Design of Experiments*, pages 207–226. Springer New York, New York, NY.

Reiff, J. and Schlegel, D. (2022). Hybrid project management–a systematic literature review. *International journal of information systems and project management*, 10(2):45–63.

Ren, J., Harman, M., and Di Penta, M. (2011). Cooperative co-evolutionary optimization of software project staff assignments and job scheduling. In *Search Based Software Engineering: Third International Symposium, SSBSE 2011, Szeged, Hungary, September 10-12, 2011. Proceedings 3*, pages 127–141. Springer.

Reynierse, J. H., Ackerman, D., Fink, A. A., and Harker, J. B. (2000). The effects of personality and management role on perceived values in business settings. *International Journal of Value-Based Management*, 13:1–13.

Rezende, A. V., Silva, L., Britto, A., and Amaral, R. (2019). Software project scheduling problem in the context of search-based software engineering: A systematic review. *Journal of Systems and Software*, 155:43–56.

Riso, D. R. and Hudson, R. (2003). *Discovering your personality type: The essential introduction to the Enneagram*. Houghton Mifflin Harcourt.

Rodríguez, D., Ruiz, M., Riquelme, J. C., and Harrison, R. (2011). Multiobjective simulation optimisation in software project management. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 1883–1890.

Sabar, N. R., Turky, A., and Song, A. (2018). A genetic programming based iterated local search for software project scheduling. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1364–1370.

Salameh, H. (2014). What, when, why, and how? a comparison between agile project management and traditional project management methods. *International Journal of Business and Management Review*, 2(5):52–74.

Saldaña-Ramos, J., Sanz-Esteban, A., García, J., and Amescua, A. (2014). Skills and abilities for working in a global software development team: a competence model. *Journal of software: evolution and process*, 26(3):329–338.

Salo, O. and Abrahamsson, P. (2008). Agile methods in european embedded software development organisations: a survey on the actual use and usefulness of extreme programming and scrum. *IET software*, 2(1):58–64.

Sánchez-Gordón, M., Rijal, L., and Colomo-Palacios, R. (2020). Beyond technical skills in software testing: Automated versus manual testing. In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, pages 161–164.

Schulze, A. and Brusoni, S. (2022). How dynamic capabilities change ordinary capabilities: Reconnecting attention control and problem-solving. *Strategic Management Journal*, 43(12):2447–2477.

Schwaber, K. and Beedle, M. (2001). *Agile software development with Scrum*. Prentice Hall PTR.

Scullard, M. and Baum, D. (2015). *Everything DiSC Manual*. John Wiley & Sons.

Shao, C. C., Kennedy, G. E., Rentas, C. M., Chen, H., and Fazendin, J. M. (2022). Leadership development among junior surgery residents: communication and perception. *Journal of Surgical Research*, 277:A18–A24.

Shen, X., Guo, Y., and Li, A. (2020). Cooperative coevolution with an improved resource allocation for large-scale multi-objective software project scheduling. *Applied Soft Computing*, 88:106059.

Shen, X., Minku, L. L., Bahsoon, R., and Yao, X. (2015). Dynamic software project scheduling through a proactive-rescheduling method. *IEEE Transactions on software engineering*, 42(7):658–686.

Shen, X., Yao, C., Song, L., Xu, J., and Mao, M. (2024). Coevolutionary scheduling of dynamic software project considering the new skill learning. *Automated Software Engineering*, 31(1):14.

Shen, X.-N., Minku, L. L., Marturi, N., Guo, Y.-N., and Han, Y. (2018). A q-learning-based memetic algorithm for multi-objective dynamic software project scheduling. *Information Sciences*, 428:1–29.

Shenhar, A. J. and Dvir, D. (2007). Project management research—the challenge and opportunity. *Project management journal*, 38(2):93–99.

Sherstyuk, O., Olekh, T., and Kolesnikova, K. (2016). The research on role differentiation as a method of forming the project team. *Eastern-European Journal of Enterprise Technologies*, 2(3 (80)):63–68.

Silva, G. F. d., Silva, L., and Britto, A. (2020). Dynamic software project scheduling problem with pso and dynamic strategies based on memory. In *Intelligent Systems: 9th Brazilian Conference, BRACIS 2020, Rio Grande, Brazil, October 20–23, 2020, Proceedings, Part I 9*, pages 79–94. Springer.

Simos, G., Constantinos, S., and Andrea, S. A. (2012). An investigation of optimal project scheduling and team staffing in software development using particle swarm optimization. In *Proceedings of the 14th International Conference on Enterprise Information Systems - Volume 2: ICEIS,*, pages 168–171. INSTICC, SciTePress.

Slowikowski, M. K. (2005). Using the disc behavioral instrument to guide leadership and communication. *AORN journal*, 82(5):835–843.

Smith, M., Polglase, G., and Parry, C. (2017). Construction of student groups using belbin: Supporting group work in environmental management. In *Pedagogic Research in Geography Higher Education*, pages 143–159. Routledge.

Snauwaert, J. and Vanhoucke, M. (2023). A classification and new benchmark instances for the multi-skilled resource-constrained project scheduling problem. *European Journal of Operational Research*, 307(1):1–19.

Söderlund, J. (2004). Building theories of project management: past research, questions for the future. *International journal of project management*, 22(3):183–191.

Soomro, A. B., Salleh, N., Mendes, E., Grundy, J., Burch, G., and Nordin, A. (2016). The effect of software engineers' personality traits on team climate and performance: A systematic literature review. *Information and software technology*, 73:52–65.

Stankūnas, M., Sauliūnė, S., Smith, T., Avery, M., Šumskas, L., and Czabanowska, K. (2012). Evaluation of leadership competencies of executives in lithuanian public health institutions. *Medicina*, 48(11):85.

Stapleton, J. (1997). *DSDM, dynamic systems development method: the method in practice*. Cambridge University Press.

Steptoe-Warren, G. (2013). *Occupational psychology: An applied approach*. Pearson Education.

Stylianou, C. and Andreou, A. S. (2013). A multi-objective genetic algorithm for intelligent software project scheduling and team staffing. *Intelligent Decision Technologies*, 7(1):59–80.

Stylianou, C., Gerasimou, S., and Andreou, A. S. (2012). A novel prototype tool for intelligent software project scheduling and staffing enhanced with personality factors. In *2012 IEEE 24th International Conference on Tools with Artificial Intelligence*, volume 1, pages 277–284. IEEE.

Sudhakar, G. P. (2016). Understanding the meaning of "project success". *Binus Business Review*, 7(2):163–169.

Sukhoo, A., Barnard, A., Eloff, M. M., Van der Poll, J. A., and Motah, M. (2005). Accommodating soft skills in software project management. *Issues in Informing Science & Information Technology*, 2.

Suri, B. and Jajoria, P. (2013). Using ant colony optimization in software development project scheduling. In *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2101–2106. IEEE.

Szwarc, E., Golińska-Dawson, P., Bocewicz, G., and Banaszak, Z. (2023). Job rotation for the competencies maintaining: A case study in it project management. In *International Symposium on Distributed Computing and Artificial Intelligence*, pages 13–23. Springer.

Szwarc, E., Golińska-Dawson, P., Bocewicz, G., and Banaszak, Z. (2024). Robust scheduling of multi-skilled workforce allocation: Job rotation approach. *Electronics*, 13(2):392.

Takeuchi, H. and Nonaka, I. (1986). The new new product development game. *Harvard business review*, 64(1):137–146.

Talib, N. A., Riaz, A., and Iqbal, M. J. (2014). Influence of national and engineering culture on team role selection. *International Journal of Technology and Design Education*, 24:91–105.

Tata, J. and Prasad, S. (2004). Team self-management, organizational structure, and judgments of team effectiveness. *Journal of Managerial Issues*, pages 248–265.

Toljaga-Nikolic, D., Petrovic, D., and Mihic, M. (2017). How to choose the appropriate project management approach? In *2017 12th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT)*, volume 2, pages 1–5. IEEE.

Towry, K. L. (2003). Control in a teamwork environment—the impact of social ties on the effectiveness of mutual monitoring contracts. *The Accounting Review*, 78(4):1069–1095.

Tuckman, B. W. (1965). Developmental sequence in small groups. *Psychological bulletin*, 63(6):384.

Turner, R. J., Huemann, M., Anbari, F. T., and Bredillet, C. N. (2010). *Perspectives on projects*. Routledge.

Twardochleb, M. (2017). Optimal selection of team members according to belbin's theory. *Zeszyty Naukowe Akademii Morskiej w Szczecinie*, 51(123):109–115.

Van Dijk, H., Van Engen, M. L., and Van Knippenberg, D. (2012). Defying conventional wisdom: A meta-analytical examination of the differences between demographic and job-related diversity relationships with performance. *Organizational Behavior and Human Decision Processes*, 119(1):38–53.

Van Knippenberg, D., De Dreu, C. K., and Homan, A. C. (2004). Work group diversity and group performance: an integrative model and research agenda. *Journal of applied psychology*, 89(6):1008.

Van Slyke, R. M. (1963). Monte carlo methods and the pert problem. *Operations Research*, 11(5):839–860.

Vanhoucke, M. and Coelho, J. (2019). Resource-constrained project scheduling with activity splitting and setup times. *Computers & Operations Research*, 109:230–249.

Vega-Velázquez, M. Á., García-Nájera, A., and Cervantes, H. (2018). A survey on the software project scheduling problem. *International Journal of Production Economics*, 202:145–161.

VersionOne (2013). 7th-annual-state-of-agile-development-survey. Technical report, VersionOne Inc.

VersionOne (2017). 11th-annual-state-of-agile-report. Technical report, VersionOne Inc.

VersionOne (2024). 17th-annual-state-of-agile-report. Technical report, VersionOne Inc.

Vishnubhotla, S. D., Mendes, E., and Lundberg, L. (2018). Designing a capability-centric web tool to support agile team composition and task allocation: a work in progress. In *Proceedings of the 11th International Workshop on Cooperative and Human Aspects of Software Engineering*, pages 41–44.

Waleed, A., Ahmad, W., Jan, M., Ali, S., Khattak, A., and Nadeem, A. (2021). The effect of team value diversity on team performance: The mediating role of relationship conflict and the moderating effects of organization citizenship behavior and leader-member exchange quality. *Management Science Letters*, 11(8):2185–2194.

Wang, M., Liu, G., and Lin, X. (2022). Dynamic optimization of the multi-skilled resource-constrained project scheduling problem with uncertainty in resource availability. *Mathematics*, 10(17):3070.

Wankhede, V. A. and Vinodh, S. (2021). Analysis of industry 4.0 challenges using best worst method: A case study. *Computers & Industrial Engineering*, 159:107487.

Wateridge, J. (1999). The role of configuration management in the development and management of information systems/technology (is/it) projects. *International Journal of Project Management*, 17(4):237–241.

Wawak, S. and Woźniak, K. (2020). Evolution of project management studies in the xxi century. *International Journal of Managing Projects in Business*, 13(4):867–888.

Weiner, M. O. et al. (2019). *The Complete Father: Essential Concepts and Archetypes*. McFarland.

Wena, F. and Lin, C.-M. (2008). Multistage human resource allocation for software development by multiobjective genetic algorithm. *The Open Applied Mathematics Journal*, 2(1).

Wheelwright, S. (1992). *Revolutionizing product development: quantum leaps in speed, efficiency, and quality*. Simon and Schuster.

Winter, B. (2015). *Agile performance improvement: The new synergy of agile and human performance technology*. Springer.

Witkowski, S. and Ilski, S. (2000). Walidacja kwestionariusza ról zespołowych: A self-perception inventory rm belbina. *Przegląd Psychologiczny*, 43(1):47–64.

Wu, G., Liu, C., Zhao, X., and Zuo, J. (2017). Investigating the relationship between communication-conflict interaction and project success among construction project teams. *International Journal of Project Management*, 35(8):1466–1482.

Wu, X., Consoli, P., Minku, L., Ochoa, G., and Yao, X. (2016). An evolutionary hyper-heuristic for the software project scheduling problem. In *Parallel Problem Solving from Nature–PPSN XIV: 14th International Conference, Edinburgh, UK, September 17-21, 2016, Proceedings 14*, pages 37–47. Springer.

Wujec, T. (2010). The marshmallow challenge. *TED Talks*.

Wysocki, R. K. (2010). *Effective software project management*. Wiley+ ORM.

Wysocki, R. K. (2019). *Effective project management: traditional, agile, extreme*. John Wiley & Sons.

Xia, X., Lo, D., Bao, L., Sharma, A., and Li, S. (2017). Personality and project success: Insights from a large-scale study with professionals. In *2017 IEEE International conference on software maintenance and evolution (ICSME)*, pages 318–328. IEEE.

Xiao, J., Ao, X.-T., and Tang, Y. (2013a). Solving software project scheduling problems with ant colony optimization. *Comput. Oper. Res.*, 40(1):33–46.

Xiao, J., Ao, X.-T., and Tang, Y. (2013b). Solving software project scheduling problems with ant colony optimization. *Computers & Operations Research*, 40(1):33–46.

Xiao, J., Gao, M.-L., and Huang, M.-M. (2015). Empirical study of multi-objective ant colony optimization to software project scheduling problems. In *Proceedings of the 2015 annual conference on genetic and evolutionary computation*, pages 759–766.

Xiao, J., Osterweil, L. J., Wang, Q., and Li, M. (2010). Dynamic resource scheduling in disruption-prone software development environments. In *Fundamental Approaches to Software Engineering: 13th International Conference*, pages 107–122. Springer.

Xu, F. and Correia, A.-P. (2024). Adopting distributed pair programming as an effective team learning activity: a systematic review. *Journal of Computing in Higher Education*, 36(2):320–349.

Yannibelli, V. and Amandi, A. (2011). A knowledge-based evolutionary assistant to software development project scheduling. *Expert Systems with Applications*, 38(7):8403–8413.

Yarramsetti, S. and Kousalya, G. (2006). An optimized event based software project scheduling with uncertainty treatment. *ARPN J. Eng. Appl. Sci.*, 10(6):2613–2620.

Yin, R. K. (2009). *Case study research: Design and methods*, volume 5. sage.

Zainal, P., Razali, D., and Mansor, Z. (2020). Team formation for agile software development: a review. *Int. J. Adv. Sci. Eng. Inf. Technol*, 10(2):555–561.

Zapotecas-Martínez, S., García-Nájera, A., and Cervantes, H. (2020). Multi-objective optimization in the agile software project scheduling using decomposition. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, pages 1495–1502.

Zhang, J., Shen, X., and Yao, C. (2023). Evolutionary algorithm for software project scheduling considering team relationships. *IEEE Access*, 11:43690–43706.

Zhu, L., Lin, J., Li, Y.-Y., and Wang, Z.-J. (2021). A decomposition-based multi-objective genetic programming hyper-heuristic approach for the multi-skill resource constrained project scheduling problem. *Knowledge-based systems*, 225:107099.

Zimmermann, A. and Trautmann, N. (2018). A list-scheduling heuristic for the short-term planning of assessment centers. *Journal of scheduling*, 21:131–142.