Tampere University

Kari Sainio

GENERATIVE ARTIFICIAL INTELLIGENCE ASSISTING IN AGILE PROJECT PAIN POINTS

Empirical study using ChatGPT

Master's Thesis
Faculty of Management and Business
Prof. Pekka Abrahamsson
Lecturer Harri Keto
August 2023

ABSTRACT

Kari Sainio: Generative Artificial Intelligence Assisting in Agile Project Pain Points
Master's Thesis
Tampere University
Master's degree Programme in Management and Information Technology
August 2023

Agile project management methods enable project work that aims to adapt to various changing conditions during the project and to control and produce the project's outputs in short iterations. In agile project management methods, the focus of traditional project management has shifted from management and administrative tasks to a kind of project coaching. Many factors influence the success of an agile project, and its various pain points must be managed. Intelligent project management can be thought of as automating its various areas in appropriate situations using artificial intelligence. In this case, especially Generative AI (GenAI) could serve as a helping hand when transitioning to intelligent project management and especially assist in managing the different pain points of projects.

In this master's thesis, a literature review is used to determine what typical challenges various agile project management methods might pose. From these challenges, a theoretical pain point model is created. As remedies for these pain points, the use of GenAI is proposed for various practical challenges in project management. In the design science research section, the capabilities of an artificial intelligence program (ChatGPT) are utilized, and usage models of GenAI artificial intelligence are presented for managing different pain points. In addition, the reliability of the information produced by ChatGPT and its use as an example to assist in the different stages of fictional project tasks are assessed. As a result, various prompt patterns are presented on how ChatGPT could be instructed to assist in managing different pain points.

As outcome, the study suggests that ChatGPT is a potential tool to assist in managing various pain points that arise in a project. Using it as a full replacement for a project manager is not yet sensible, but in suitable assisting, guiding, and inspiring tasks, generative artificial intelligence can facilitate work. The study also found that ChatGPT's knowledge of the challenges of agile methods and various frameworks is reliable. Therefore, it could be used in various project management pain points for review, preparation, and coaching. However, as a final note, it can be suggested that it might be wise to approach all material produced by ChatGPT with caution, and it is advisable to also use knowledgeable individuals to evaluate and review the content and reliability of the information produced by it.

Keywords: Agile Project Management, Agile Project Pain Points, ChatGPT, Generative AI

The originality of this thesis has been checked using the Turnitin OriginalityCheck service.

TIIVISTELMÄ

Kari Sainio: Generative Artificial Intelligence Assisting in Agile Project Pain Points
Diplomityö
Tampereen yliopisto
Johtamisen ja tietotekniikan maisterikoulutus
Elokuu 2023

Ketterät projektinhallintamenetelmät mahdollistavat projektityöskentelyn, jossa pyritään sopeutumaan projektin aikana erilaisiin muuttuviin olosuhteisiin ja kontrolloimaan sekä tuottamaan projektin tuotoksia lyhyissä iteraatioissa. Ketterissä projektinhallintamenetelmissä perinteisen projektinhallinnan painopiste on siirtynyt johtamisesta ja hallinnallisista töistä eräänlaiseksi projektin valmentamiseksi. Ketterän projektin onnistumiseen vaikuttavat monet tekijät ja sen erilaisia kipupisteitä pitää pystyä hallitsemaan. Älykkään projektinhallinnan voidaan ajatella olevan sen eri osa-alueiden automatisointia sopivissa tilanteissa tekoälyä hyödyntäen. Erityisesti Generatiivinen tekoäly (GenAI) voisi tässä tapauksessa toimia auttavana kätenä älykkääseen projektinhallintaan siirryttäessä sekä erityisesti avustaa projektien erilaisten kipupisteiden hallinnassa.

Tässä diplomityössä selvitetään kirjallisuuskatsauksen avulla millaisia tyypillisiä haasteita erilaiset ketterät projektinhallintamenetelmät saattavat aiheuttaa. Näistä haasteista luodaan teoreettinen kipupistemalli. Kipupisteiden lääkkeinä esitetään generatiivisen tekoälyn hyödyntämistä erilaisissa projektinhallinnan käytännön haasteissa. Suunnittelutieteellisessä tutkimusosiossa hyödynnetään generatiivisen tekoälyohjelman (ChatGPT) kyvykkyyttä ja esitetään käyttömalleja eri kipupisteiden hallitsemiseksi. Tämän lisäksi arvioidaan ChatGPT:n tuottaman tiedon luotettavuutta sekä sen käyttöä esimerkkinä kuvitteellisten projektitehtävien eri vaiheiden apuna. Lopputuloksena esitetään erilaisia kehotemalleja, joiden avulla ChatGPT:tä voitaisiin pyytää avustamaan projektin erilaisten kipupisteiden hallinnassa.

Johtopäätöksenä tutkielmassa esitetään, että ChatGPT on potentiaalinen työkalu avustamaan erilaisia projektissa ilmenevien kipupisteiden hallinnassa. Sen käyttö projektipäällikön täysimittaisena korvaajana ei vielä ole järkevää, mutta soveltuvissa avustavissa, opastavissa sekä inspiroivissa tehtävissä, generatiivinen tekoäly voi helpottaa työtä. Työssä havaittiin myös, että ChatGPT:n tietämys ketterien menetelmien haasteista sekä erilaisista viitekehyksistä on luotettavaa ja näin ollen sitä voitaisiin käyttää erilaisissa projektinhallinnan kipupisteissä katselmoimaan, esivalmistelemaan ja valmentamiseen. Lopputulemana voidaan kuitenkin esittää, että kaikkeen ChatGPT:n tuottamaan materiaaliin voi olla hyvä luottaa varauksella ja on suositeltavaa käyttää lisäksi asiantuntevia henkilöitä arvioimaan sekä katselmoimaan sillä tuotettujen tietomateriaalien sisältöä sekä luotettavuutta.

Asiasanat: ChatGPT, generatiivinen tekoäly, ketterä projektinhallinta, ketteryyden kipupisteet

Tämän julkaisun alkuperäisyys on tarkastettu Turnitin OriginalityCheck –ohjelmalla.

PREFACE

Working on ChatGPT and using it for assisting in agile project management during its early days has been interesting journey. In year 2023 Generative AI seems to have lots of hype and conversation around how to use ChatGPT in various manners and other Generative AI tools. The scene has gained popularity and expertise. It seems the world has stepped forward again since the invention of Web browser and the Internet how we use ITC. As seen the days of the Internet rising since 1990's, I am looking forward to seeing what happens in AI – hopefully only in good ways.

I thank professor Pekka Abrahamsson for idea to extend original topic to include GenAI and especially ChatGPT how to utilize that in various projects management challenges. I also thank my lovely family for supporting my studies as lifetime learner. It has been interesting time to study when Covid-19 has been challenging the whole world and has changed how we work and operate in a new and probably more innovative way. Perhaps in this manner GenAI can assist everyone to make world a better place and make us learn more from the pre-trained history, civilization, and humanity and it will be not used for bad purposes.

Vantaa 8.8.2023

Kari Sainio

CONTENTS

1. INTRODUCTION 1
   - 1.1 Motivation 2
   - 1.2 Research questions 2
   - 1.3 Scope of work 2
   - 1.4 Research methods 3
   - 1.5 Structure of thesis 3
2. THEORETICAL FRAMEWORK 4
   - 2.1 Predictive projects 4
   - 2.2 Small scale agile projects 5
     - 2.2.1 Scrum 5
     - 2.2.2 Extreme Programming (XP) 6
     - 2.2.3 Kanban 7
     - 2.2.4 Lean Software Development (LSD) 8
   - 2.3 Large scale agile projects 9
     - 2.3.1 Scalable Agile Framework (SAFe) 10
     - 2.3.2 Large Scale Scrum (LeSS) 12
     - 2.3.3 Disciplined Agile (DA) 14
     - 2.3.4 Other agile frameworks 17
   - 2.4 Generative Artificial Intelligence in project management 17
     - 2.4.1 Generative Artificial Intelligence 18
     - 2.4.2 Large Language Models 19
     - 2.4.3 Google Bard 21
     - 2.4.4 Bing chat 22
     - 2.4.5 Baidu Ernie 22
   - 2.5 ChatGPT 23
     - 2.5.1 Training data 24

2.5.2 GPT-4 24
2.5.3 Limitations and Challenges 25
2.5.4 Applications and Use Cases 25
3. RESEARCH MODEL: THE AGILE PAINT POINT MODEL 26
3.1 Agile pain points in literature 26
3.2 Construction of pain point model 31
3.2.1 Requirements management 33
3.2.2 Stakeholder and management support 34
3.2.3 Role definition 34
3.2.4 Redundancy 35
3.2.5 Agile process understanding 35
3.2.6 Adaptability 36
3.2.7 Change resistance 36
3.2.8 Maintaining agile way of working 37
3.2.9 Work estimation 37
3.2.10 Technical knowledge 38
4. RESEARCH METHOD: DESIGN SCIENCE RESEARCH 39
4.1 Problem Identification and Objectives Definition 40
4.2 Development 40
4.3 Demonstration and Evaluation 42
5. RESULTS 44
5.1.1 Requirements management 45
5.1.2 Stakeholder and management support 47
5.1.3 Role clarification 49
5.1.4 Redundancy analysis 52
5.1.5 Agile process understanding 54
5.1.6 Change resistance 57
5.1.7 Adaptability understanding 60

5.1.8 Maintaining agile way of working 61
5.1.9 Effort estimation 63
5.1.10 Technical knowledge 65
6. DISCUSSION 69
6.1 Theoretical implications 69
6.2 Practical implications 73
7. CAPABILITIES OF CHATGPT 76
7.1 Answers to research questions 77
7.2 Limitations of research 78
7.3 Usage of ChatGPT 79
7.4 Future research opportunities 80
REFERENCES 81
APPENDIX A: PROMPT TEST ROUND 1 92
APPENDIX B: PROMPT TEST ROUND 2 137
APPENDIX C: PROMPT TEST ROUND 3 178
APPENDIX D: PROMPT CHEAT SHEET 254

LIST OF FIGURES

Figure 1. Scrum framework [18] 6

Figure 2. Kanban board [12][22] 8

Figure 3. SAFe 4.5 framework (FULL) [30] 10

Figure 4. LeSS principles [33] 12

Figure 5 LeSS Framework [33] 13

Figure 6. LeSS Huge [33] 14

Figure 7. Disciplined Agile [34] 15

Figure 8. Organizing a large agile team in DA [35] 16

Figure 9. Language model sizes [52] 19

Figure 10. Transformer architecture [57] 21

Figure 11. The framework of ERNIE 3.0 Titan [45] 23

Figure 12. Agile project challenges [76] 29

Figure 13. Paint points 33

Figure 14. Research process 39

Figure 15. Example of prompt [98]. 41

Figure 16. Roles specified by ChatGPT 51

LIST OF TABLES

Table 1. Classification of agile pain points. *32*

Table 2. Prompt pattern classification. *44*

Table 3. Requirements creation pattern *45*

Table 4. Steering group pattern *47*

Table 5. Role clarification pattern *49*

Table 6. Redundancy analysis pattern *52*

Table 7. Agile process coaching pattern *55*

Table 8. Change resistance pattern *58*

Table 9. Adaptability management pattern *60*

Table 10. Agile way of working pattern *62*

Table 11. Effort estimation pattern *63*

Table 12. Technical knowledge analysis pattern *65*

Table 13. Contributions in relation to literature *70*

Table 14. Proposals for using ChatGPT *73*

Table 15. General contextual prompt pattern *254*

Table 16. Prompt pattern cheat sheet for agile pain points *254*

LIST OF SYMBOLS AND ABBREVIATIONS

AI	Artificial Intelligence
APM^1	Agile Project Management
APM^2	Agile Product Management
ARM	Agile Requirements Management
ASD	Agile Software Development
AUP	Agile Unified Process
AWS	Amazon Web Services
BERT	Bidirectional Encoder Representations from Transformers
CDK	Cloud Development Kit
ChatGPT	Generative AI system by OpenAI
CoAI	Conversational AI
DA	Disciplined Agile
DAD	Disciplined Agile Delivery
DevOps	Development and Operations
DSDM	Dynamic System Development Method
DSR	Design Science Research
FDD	Feature Driven Development
GenAI	Generative AI
GPT	Generative Pre-trained Transformer
GT	Grounded Theory
ISE	Intelligent Software Engineering
LeSS	Large-Scale Scrum
LLM	Large Language Model
LSD	Lean Software Development
ML	Machine Learning
MIP	Mixed-Integral Programming
MoSCoW	Prioritization technique
NLP	Natural Language Processing
PMI	Project Management Institute
PO	Product Owner
RACI	Responsible, Accountable, Consulted, Informed
RLHF	Reinforced Learning by Human Feedback
SAFe	Scalable Agile Framework
SM	Scrum Master
SWOT	Strengths, Weaknesses, Opportunities, Treats
WoW	Way of Working
XP	Extreme Programming

1. INTRODUCTION

Project management often encounters several challenges [1]. Agile project management (APM) is an approach to control that relies on empirical observations within the project [2]. Its focus is on sustaining successful project implementation despite changes in the environment [3]. As a more reactive methodology, it may encounter various challenges throughout the project life cycle. Furthermore, agile project management challenges may be classified to project, team, individual and task level [4]. APM may include alterations to project scope, changes in resources, and other shifts in the business or organizational landscape, which can create a range of management obstacles [2]. Consequently, modern project management requires a great deal of skill and effort to address these issues [1].

Moreover, challenge can be scope creep, where projects expand beyond their original objectives, causing time and budget overruns. Resource management can be another challenge, with unexpected changes in personnel or material resources leading to delays. Additionally, unclear communication among team members can lead to confusion and inefficiencies. Constant shifts in the business or regulatory landscape also add to the complexity, necessitating frequent adjustments in project direction. Lastly, stakeholder management can be difficult, as varying interests and expectations may conflict with project goals. These kinds of challenges can be called pain points which are examples that must pay attention in strategic and adaptive project management practices to ensure success.

Over time, Artificial Intelligence (AI) has developed to include systems and methods capable of processing natural language. Conversational AI (CoAI) bots, such as E-Commerce Customer Service Bots, Amazon Echo Alexa, and Virtual Personal Assistants, have been widely used [5]. Lately, more advanced conversational systems such as ChatGPT have been developed, which can conduct conversational discussions and provide answers to various questions and problems users face [6]. Generative AI (GenAI) refers to deep learning models capable of creating high-grade text, imagery, and other types of content, in accordance with the training data they have been provided with [7]. Utilizing conversational and generative AI these systems could potentially automate or assist project management tasks and provide suggestions to manage common pain points found in typical agile projects. The aim of this thesis was to examine how GenAI, using ChatGPT as an example can be used to tackle agile project pain points.

1.1 Motivation

Agile project management (APM[1]) has become trendy way of managing different IT related development projects such as software development for various systems. APM requires lots of discipline from the project organization and the project manager. To help in reaching projects their target help of AI becomes interesting topic to research. From the scientific point of view AI includes several approaches and techniques, such as machine learning, machine reasoning, and robotic [8]. As the capabilities of AI is increasing an interest has raised whether some parts of today's project management can be automated, or some tasks can be hand overed to AI to take responsibility of and by that way reduce typical project manager's work and even help in difficult situations [9].

1.2 Research questions

The main objective of this study is to explore how AI can be employed to address the challenges and pain points associated with agile project management, irrespective of its scale. Specifically, AI solutions offering GenAI could assist in daily operations by offering coaching, recommendations, and guidance. In this context, interactive and generative artificial intelligence, such as ChatGPT, presents a compelling avenue to understand how it can aid in navigating regular routines and challenging scenarios, thereby addressing common agile project pain points documented in existing literature.

RQ1: What are the typical pain points of agile project?

RQ2: How GenAI could guide to overcome typical pain points of agile projects?

RQ3: Can GenAI used in project management assistance such as planning, reporting, and estimating?

1.3 Scope of work

The scope of the work covers theoretical background for agile project management, use of generative artificial intelligence and how those can be utilized in agile project management. Design science research (DSR) is made for the current version of

ChatGPT system (GPT4, spring 2023) and the outcome is reflected to the theoretical knowledge of today's methodologies and challenges in the APM. Predictive project management styles such as waterfall is left out of the scope of this study [10]. Also, this research concentrated on Agile methodology where viewpoint can range from the developer through project manager to any project staff.

1.4 Research methods

The theoretical framework for general agile project management is established through literature research [11]. A model identifying pain points is created from the findings of this literature research. The empirical portion of the study is constructed based on design research, which focuses on the application of GenAI. In this context, the usage of ChatGPT is exemplified, aligning ChatGPT with both theory and practical application. The objective is to offer a reasonable understanding of how GenAI, particularly ChatGPT, can be effectively used in project management related challenges, while also acknowledging its limitations and the necessity for human oversight.

1.5 Structure of thesis

The structure of this thesis encompasses six major sections. The initial chapter delineates the motivation behind the research and outlines the research objectives. This is followed by the theoretical section which introduces common agile methodologies, both at small and large scales, and discusses the mechanism of artificial intelligence employed in this study. Moreover, GenAI and ChatGPT is described. The third major chapter illustrates the research model pain points associated with agile projects. The fourth segment embarks on the design science research, emphasizing the advantages of using GenAI to alleviate typical pain points in agile projects. The penultimate chapter, chapter five, opens a discourse on the research findings. Chapter six contains discussion about theoretical and practical impacts. Conclusively, chapter seven addresses the research questions, summarizes the study, and speculates on potential future research in this domain.

2. THEORETICAL FRAMEWORK

According to study 50 % of the project uses hybrid methods, agile 33% and traditional 15% [12]. In agile software development number of practices cover areas such as: requirements, design, modelling, coding, testing, planning, risk management, process, and quality [12]. Agile software development itself has been defined in manifesto [13]. It defines individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation and responding to change over following a plan [13] [14]. Agile development projects can be small scale or large scale and may require different approaches. Scaling of agile has been identified to be challenge due to variety between subsystems and teams, lifecycle management in of shorter- and longer-term development, maintenance burden of old legacy systems in contrast to new development and agile requirement process management [15].

Intelligent software engineering (ISE) is an application containing intelligent techniques to software engineering where techniques can explore data for knowledge discovery. This knowledge can be used for reasoning, learning, planning, natural language processing (NLP), perception or supporting decision making [16]. In general, different AI applications, and Conversational AI (CoAI) could be considered one of the intelligent software engineering techniques. CoAI assistants (chatbots) are systems that use conversational dialogue to accomplish one or more tasks [17]. As intelligent software engineering is interesting, interest may rise to intelligent project management.

The following chapters give examples of the small- and large-scale agile frameworks that are known to be used in the agile software development process. In addition, literature research is performed how intelligent techniques could be applied to agile project management (APM). The bias of intelligent techniques is set on GenAI or CoAI and their capabilities.

2.1 Predictive projects

The Project Management Institute refers to traditional project management as "predictive project management," which contrasts with agile approaches [3]. The term "plan-driven" is often used to characterize non-agile projects because they focus on developing a comprehensive plan upfront and then executing it throughout the project lifecycle [3]. Although similarities in traditional project a.k.a. predictive projects and agile projects may exist this thesis is concentrating in agile project management and predictive project problems are left for smaller research. The following chapters describe different agile methods and frameworks that may overcome the descriptive project related challenges.

2.2 Small scale agile projects

Scrum, XP, Kanban, and Lean Software Development as example of agile framework can be considered originally as small-scale agile projects. The following chapters gives an overview of these frameworks and their properties.

2.2.1 Scrum

Scrum has been introduced in 2001 as a better way of solving complex problems at team level. A Scrum team contains different members: A Product Owner (PO), a Scrum Master (SM) and team members who are focusing on development. Scrum itself is an empirical process, where team has much decision authority of their work. Basic idea is to plan work in small iterations (sprints) and manage tasks through backlog. Team has authority to estimate and commit to the planned tasks. PO is responsible for feeding requirements called stories to team, prioritize, and accept the work done by team members. Daily routines are facilitated by SM and traditional project manager role does not exists. Scrum is repeating development process starting with sprint planning, ending to sprint review, and delivering the software or product at the end of each sprint. [18]

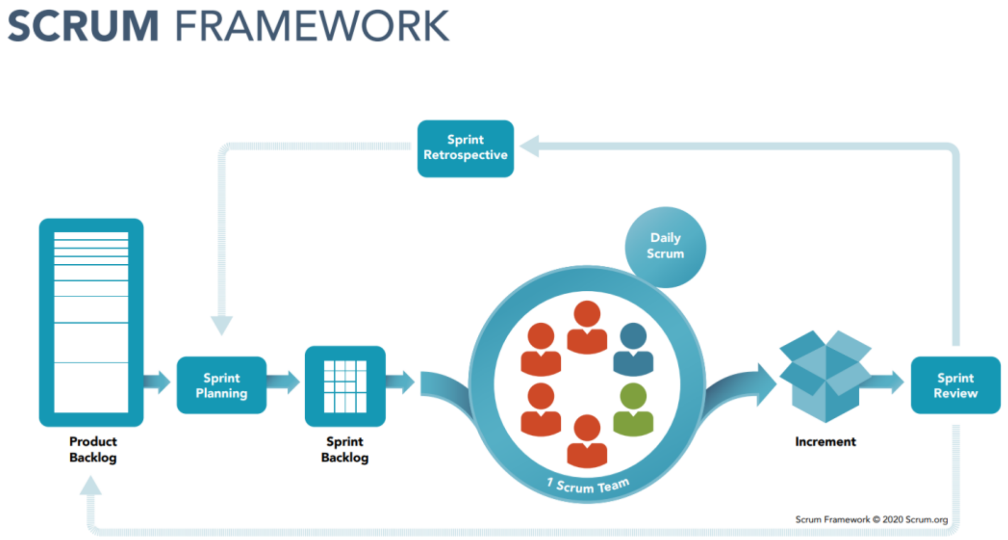


Figure 1. Scrum framework [18]

Figure 1 shows typical lifecycle of Scrum. Requirements are placed on Product Backlog where they are prioritized based on their business value. For each iteration called sprint the most important requirements (stories) are picked to sprint backlog. Sprint backlog is the work to be done in the sprint and its workload is estimated by the team. Development team is committed to do the tasks in sprint backlog during the following sprint. SM takes care of facilitating each sprint planning and sprint review meetings. During the execution of each sprint daily stand-ups are held. The intention with daily stand-ups is to follow the progress and do necessary adjustments that all tasks during sprint execution will be done accordingly. Daily stand-ups will also bring up any problems team may have that need further discussion. Stand-ups are held short and basically handles topics such as what has been finished, checking any problems and what will be done next.[19]

According to Kadenic, Koumaditis and Junker-Jensen agile teamwork performance is a consistent theme and subject of interest in the research community [20]. Also, team capability and customer involvement are the key factors for successful projects when measured by time, cost, and customer satisfaction [20].

2.2.2 Extreme Programming (XP)

According to Jeffries, Anderson, and Hendrickson:” Extreme Programming is a discipline of software development with values of simplicity, communication, feedback, and courage. We focus on the roles of customer, manager, and programmer and accord key rights and responsibilities to the people in those roles.” [21]. XP defines roles for the customer, the manager, and the programmer. The customer defines the stories that needs to be developed, implemented, and define testing that system works. The customer has also critical responsibility to provide the most valuable features that has the highest business value over delivery time. The programmer role is to analyze, design, test, program, and integrate the system. They also estimate the difficulty of all stories and maintain the understanding what stories can be delivered to the customer. The manager role connects the customer and developers into operational entity. Intention is to make the process smoother and not to create the process. The manager and customer have rights to the development work by knowing what can be published and when, the outcome of each week, to see running systems, to change content and priorities, and to be informed of any schedule changes. [21]

In XP, the developers follow certain practices such as straightforward design, refactoring, collective code ownership, and pair programming.

2.2.3 Kanban

Kanban (看板, かんばん) is originally a Japanese noun meaning a signboard. An example of this board is presented in Figure 2. Kanban framework contains four parts [22]:

- Visualisation of work as Kanban board is divided into columns representing the state of the task (Figure 2).
- Work in Progress (WIP) controls the work amount.
- Focus on the Flow which can be done by limiting the WIP and to avoid any bottlenecks in the system.
- Continuous Improvement by eliminating problems within the system to achieve shorter lead times, improving quality and have more coherent flow.

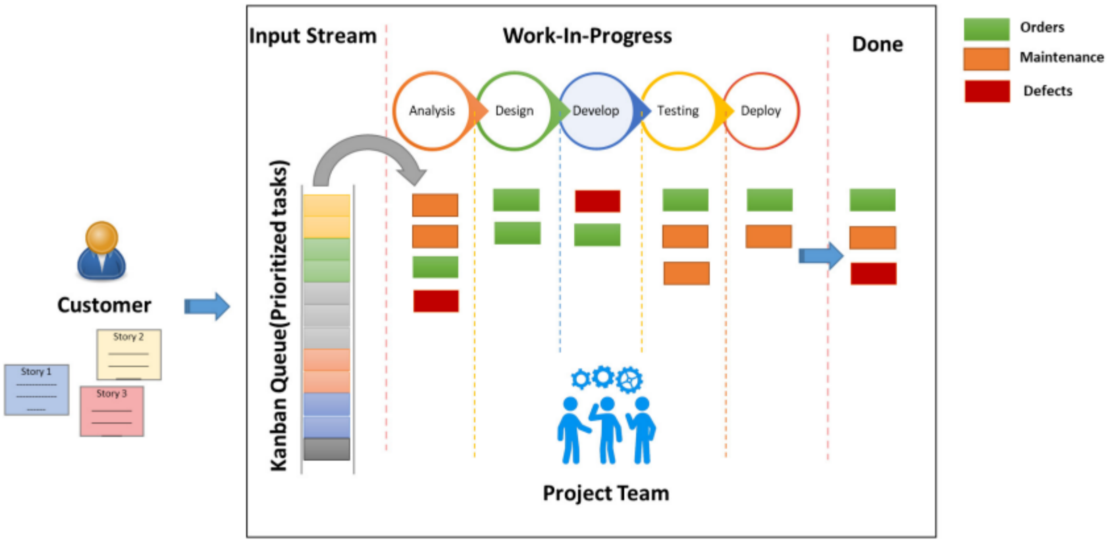


Figure 2. Kanban board [12][22]

In Kanban, teams communicate using simple tool provided information on what is needed to be done. The concept of Kanban is to work Just In Time (JIT) [22][12]. Kanban board has three phases:

- To Do: contains all tasks that are to be done, this list can be prioritized,
- In Progress: contains all tasks that are under work in progress,
- Done: shows all tasks that have been completed.

Kanban can spot bottlenecks that blocks the flow and productivity can be measured based on lead time. Kanban also provides a clear visualization of the assigned tasks for each team member. [22]

2.2.4 Lean Software Development (LSD)

The term “lean” was originally used in production management and later in product development in MIT during the mid-1980s [23]. Applying lean principled to software development is almost has been introduced almost the same time [23]. Lean in software contains seven principles which can be translated into agile practices [24]:

1. Eliminating waste by abandoning anything that does not add value to a product and is not what customer wants.

2. Amplify learning is the best approach to improve a software development environment as it ensures continuous adaptation to evolving user needs and technologies, fostering quicker feedback loops, collaboration, risk mitigation, skill development, and enhanced customer satisfaction.
3. Decide as late as possible as future is unpredictable and locking into some decisions may have some facts later instead of speculation.
4. Deliver as fast as possible as the discovery cycle is critical for learning: Design, implement, feedback, and improve. These cycles should be short that the customers get what they need now and there is place for corrections in later cycles.
5. Empower the team as involving the developers is fundamental to achieving excellence and they have the best technical understanding.
6. Building integrity in as software with integrity has a coherent architecture, usability is high and fits for its purpose, and code is maintainable.
7. See the whole as complex systems requires a deep expertise in many diverse areas and may cause local optimization which does not maximize the overall performance.

For especially eliminating waste Poppendieck has an opinion: “Waste is anything that the customers do not value. If you're doing something and in the end the customers don't think it is important for them, then it is waste. There are lots of ways to look at waste. One is: partially done work is waste. Just like in manufacturing, inventory is waste; in software development partially done work is waste. Also, extra features you don't need right now is waste; stuff that causes delays is waste; things that get in the way of a rapid flow of product are waste. Because customers, when they have a problem, they want the problem solved now, and stuff that delays that is waste.” [25]. Term Done means it has been coded, tested, documented, and integrated [25]. By following practicalities above lean major principle is doing everything to the end with added customer value, and not spending time for anything else.

2.3 Large scale agile projects

While agile software development has been originally developed to serve one teamwork the larger projects or organizations may require several teams working together. For this purpose, several different scalable frameworks have been introduced. In the following chapters some examples of the famous framework will be briefly presented. Scalable

Agile Framework (SAFe), Large Scale Scrum (LeSS) and Disciplined Agile (DA) each provides methodologies to up-scale basic agile team process to be applied for several teams in larger organizations [26]–[28].

2.3.1 Scalable Agile Framework (SAFe)

According to research Scalable Agile Framework (SAFe) has become popular due to successful marketing by Scaled Agile Inc and several companies have taken that into use [29]. The idea behind SAFe is to combine the power of Agile with Lean product development and systems thinking by following Lean-Agile principles and values, which guide the roles, responsibilities, artifacts, and activities required to gain better business outcomes [30]. SAFe is comprehensive agile framework and in full size it crosses the whole organization. Figure 3 shows the full configuration of SAFe version 4.5 including different agile roles, *functions* and enterprise level program management starting from team level towards portfolio level [30].

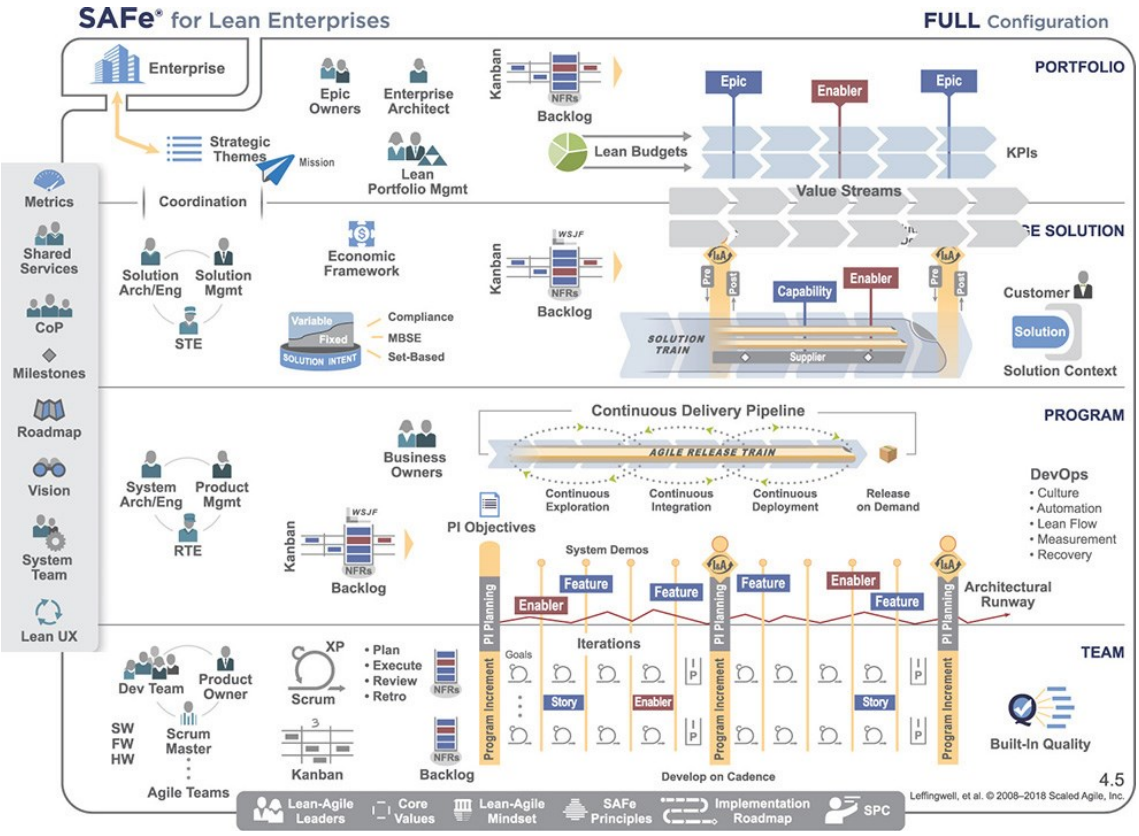


Figure 3. SAFe 4.5 framework (FULL) [30]

SAFe defines four configurations: essential, portfolio, large, and full which can be applied according to the organization size [31]. SAFe is based on the following principles [32]:

- Taking an economic view: Delivering early and often supports agile principle to deliver value for the customer much earlier than in traditional waterfall project. Furthermore, operating with lean budgets and guardrails, understanding solution economic trade-offs, leveraging supplies, sequencing jobs for maximum benefit are applied as economic framework.
- Applying systems thinking: In SAFe, systems thinking is applied to the system under development, as well as to the larger system—the Solution, and the entire enterprise.
- Assuming variability; preserve options: By maintaining multiple requirements and design options for a longer period in the development cycle, developers can adapt to Inevitable change.
- Building incrementally with fast, integrated learning cycles: This principle suggests building systems in a series of small, fully integrated increments, each of which can be run through a full set of tests, ensuring that all components work together.
- Basing milestones on objective evaluation of working systems: Integration points provide objective milestones at which to evaluate the solution throughout the development life cycle.
- Visualizing and limiting WIP, reduce batch sizes, and manage queue lengths: To achieve the shortest sustainable lead time, Lean enterprises strive for a state of continuous flow, which improves the delivery of value through the Value Stream.
- Applying cadence, synchronize with cross-domain planning: Cadence creates predictability and provides a rhythm for development. Synchronization causes multiple perspectives to be understood, resolved, and integrated at the same time.
- Unlocking the intrinsic motivation of knowledge workers: Lean enterprises respect the knowledge workers' desire for mastery, autonomy, and purpose, and they understand that people are more engaged and productive when they're doing work they care about.
- Decentralizing decision-making: Decentralizing decision-making enables faster, more localized decision-making, and it improves the speed and quality of the implementation.

- Organizing around value: The final principle is to understand the full value stream and organize around it. This principle suggests that reorganizing around value can eliminate functional silos, enable fast flow, and better deliver value.

2.3.2 Large Scale Scrum (LeSS)

Large Scale Scrum (LeSS) is Scrum applied to many feature teams working together on a common goal and on one product. LeSS itself defines rules but those are minimalistic and do not define how LeSS should be applied. To support this, LeSS is based on ten principles (Figure 4) [33]:

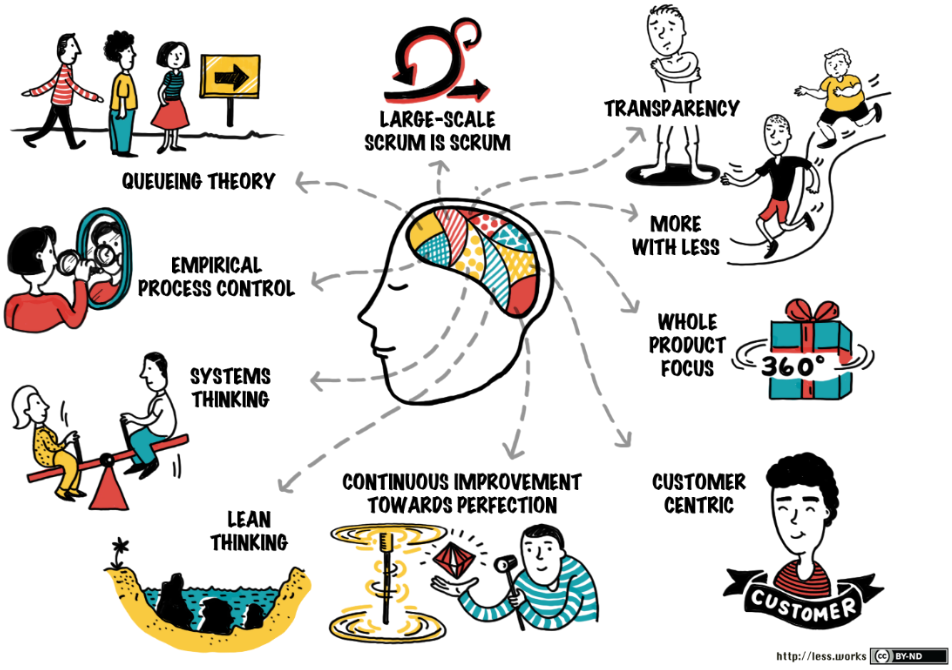


Figure 4. LeSS principles [33]

- Large-Scale Scrum is Scrum is about applying basic Scrum principles in large scale as simply as possible,
- transparency for teamwork using short feedback loops and clear definition of done (DoD)
- more with less by defining the barely sufficient needed for large-scale development. Idea is to avoid adding additional roles, artifacts, or processes, as these additions are considered as drawbacks,
- whole product focus by using one Product Backlog (PB), Product Owner (PO), one sprint and one shippable product,

- customer centric by focusing to learn customer's real problems and solve those,
- continuous improvement towards perfection is based on lean thinking,
- lean thinking by respecting the people and continuously challenge-the-status-quo by having improvement mindset,
- systems thinking by understanding the whole system and not part of it by avoiding concentrating on local optimization,
- empirical process control by constantly inspect and adapt inspect and adapt the product, processes, behaviours, organizational design, and practices to evolve in situationally appropriate ways,
- and queueing theory where the idea is to focus on value throughput by adapting different mechanisms to control cycle times without overburdening the people.

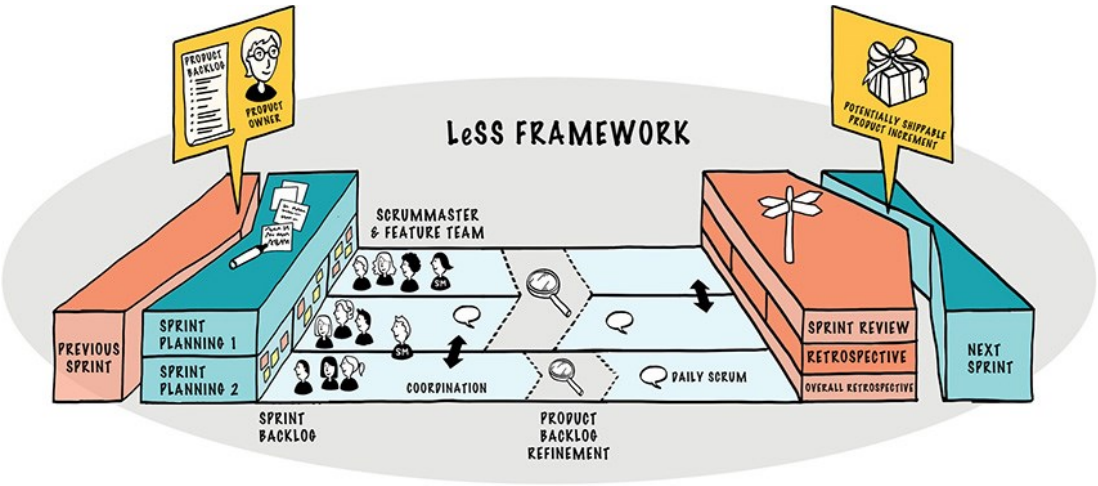


Figure 5 LeSS Framework [33]

Figure 5 shows LeSS framework which can be scaled as normal LeSS for 2-8 teams and LeSS Huge for 8+ teams. Both LeSS sizes are based on one product owner, one backlog and one common sprint and one shippable product. The LeSS framework elements corresponds the elements one-team Scrum. However, story used in Scrum team focuses on the flow of teams through a sprint and not for the flow of item. Furthermore, LeSS promotes two sprint planning where the sprint planning one is common for all teams and the sprint number two is done within a team. The sprint end to team level retrospective followed by overall retrospective with limited number of participants. The overall retrospective is held after the sprint has ended. LeSS promotes also different coordination and refinement meetings during the sprint [33].

In LeSS Huge additional roles are introduced and the different requirement areas are created. The idea is that teams split for feature areas still work in the same sprint. The elaboration of LeSS Huge can be seen in Figure 6. The basic principles stay same in LeSS and LeSS Huge [33].

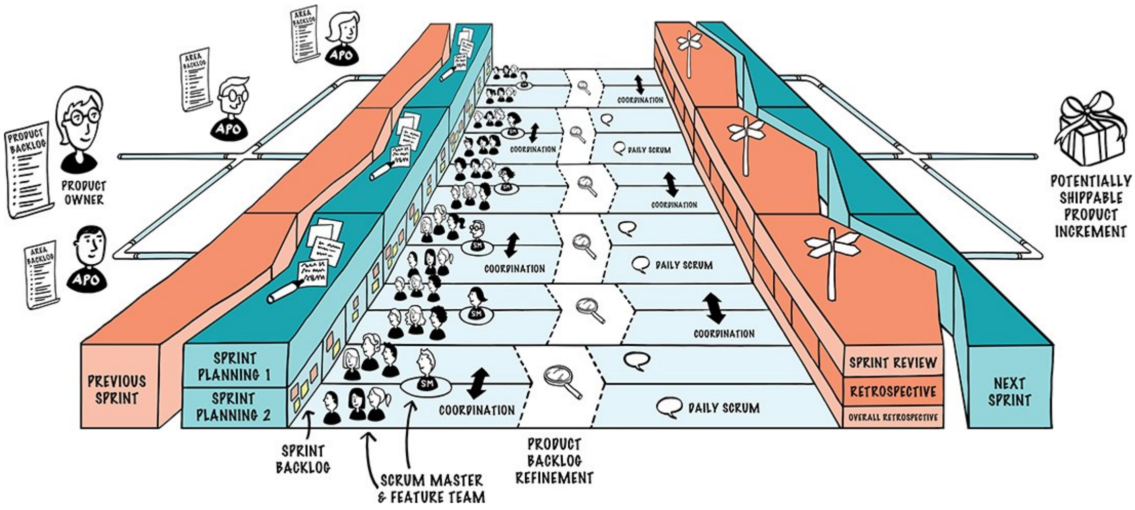


Figure 6. LeSS Huge [33]

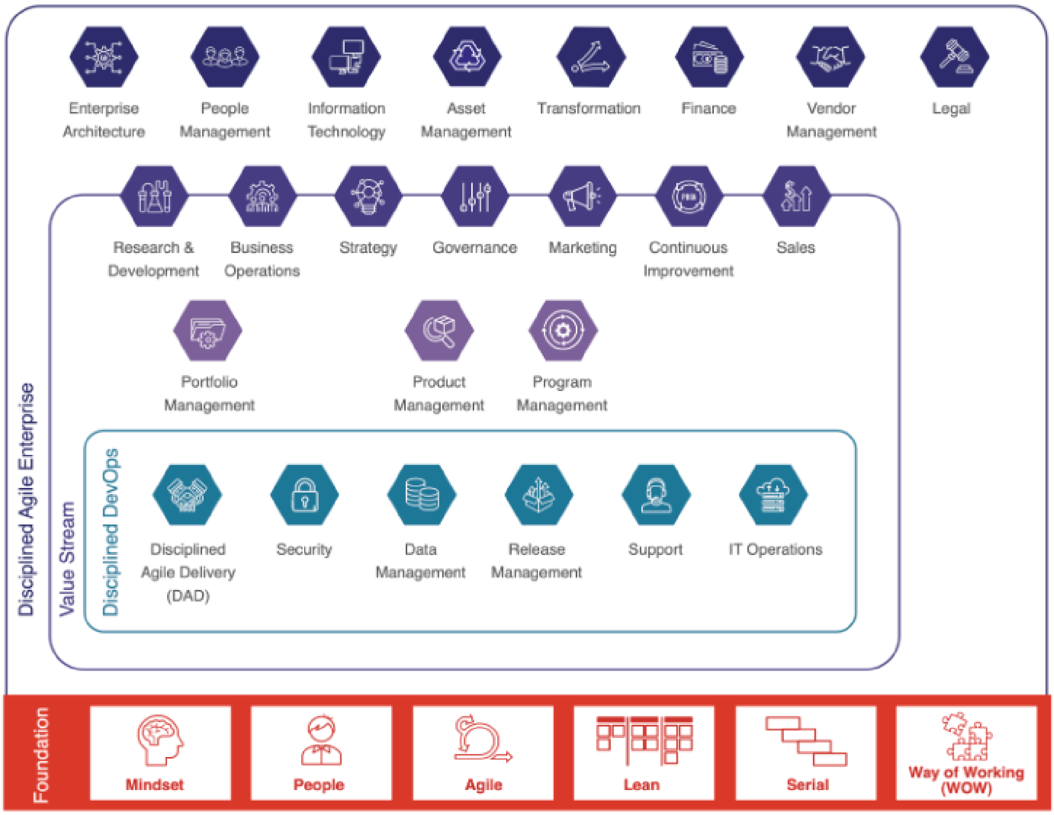
2.3.3 Disciplined Agile (DA)

Disciplined Agile (DA) is considered as a tool kit. Typically, an agile framework is presenting a collection of “best practices”. These practices are contextual and may not work well in all situations. The idea of DA is to be process-decision based tool kit that can let its users to choose own Way of Working (WoW) that fits-for-purpose [34].

DA addresses four views [34]:

1. Mindset. DA builds on the foundations of agile and lean to address enterprise realities.
2. People. DA describes the roles, responsibilities, and team structures you should have in place.
3. Flow. DA describes the dynamic aspects of processes via lifecycle and workflow diagrams.
4. Practices. DA outlines the techniques that move your team forward, using straightforward goal diagrams that provide a high-level picklist of practices.

Figure 7 shows different “areas” of DA by defining Disciplined Agile Enterprise (DAE), Value Stream, Disciplined DevOps, and Foundation [34].



©Project Management Institute. All rights reserved.

Figure 7. Disciplined Agile [34]

For many organizations Scrum is the starting point when moving into agile development [35]. Other methods overlaps and conflicts with terminology which may confuse agile practitioners, as well as stakeholders not inside of the agile development itself [35]. According to reference [28] Disciplined Agile Delivery (DAD): “Is an enterprise-aware hybrid software process framework”. DAD specifies Agile Scaling Model (ASM) that is a contextual framework to adopt and tailor agile strategies for an agile software development team [28]. ASM defines three process categories [28]:

- Core agile development that covers the basic agile methods such as Scrum, XP, and Agile Modelling (AM). Value-driven lifecycle is characterized by high-quality potentially shippable product released on regular basis on collaborative and self-organising team.

- Agile delivery defines methods including DAD process framework and Harmony/ESW to address the full delivery lifecycle from project initiation to production.
- Agility@scale is disciplined agile development where scaling factors apply. These scaling factors come from team size, geographical and organisational distribution, regulatory compliance, cultural or organizational complexity, technical complexity, and enterprise disciplines.

The DAD process framework has several characteristics: People first, Learning oriented, Agile, Hybrid, IT solution focused, Goal-driven, Delivery focused, Enterprise aware, Risk and value driven and Scalable [28]. Figure 8 shows the shows the organizational structure to support coordination within the program in DAD [35].

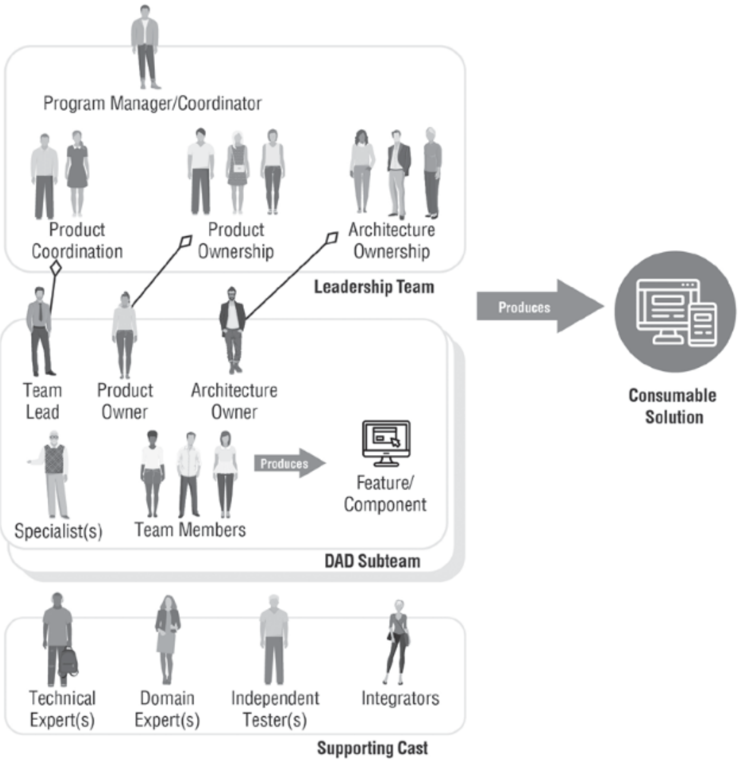


Figure 8. Organizing a large agile team in DA [35]

2.3.4 Other agile frameworks

For the entire project a single approach is not necessary. To achieve goals project often combines elements of different life cycle. In a hybrid approach a combination of predictive, iterative, incremental, and/or agile approaches exist. [10]

In addition to SAFe, LeSS and DA other agile framework exists. As an example of such frameworks are PRINCE2 Agile [36], Scrum@Scale [37] , etc exist. All these frameworks follow the original agile principles and basically are adaptation or hybrid versions to the original agile manifesto and traditional project management. However, according to a study [10] hybrid, traditional, and agile project approaches showed similar levels of success in meeting the triple constraint goals of budget, schedule, and scope/quality. There was no significant statistical difference observed among these three approaches in terms of achieving the desired targets for budget, schedule, and scope/quality [10].

2.4 Generative Artificial Intelligence in project management

Gartner predicts that by 2030, 80% of current project management tasks will be eliminated as artificial intelligence becomes more prevalent in the field [38]. Nieto-Rodriguez et al. list six items in their article where Artificial Intelligence (AI) could change the project management [9]:

- Better selection and prioritization of projects that bring the most value to the organisation,
- support for the project management office (PMO) to help organisations streamline and optimize the role of the PMO.
- improved, faster project definition, planning, and reporting,
- virtual project assistants such as ChatGPT,
- advanced testing systems and software,
- and a new role for the project manager as shifting away from administrative work.

According to Prifti the use of AI in project management has increased the productivity of project managers [39]. However, AI cannot replace project managers [39]. Furthermore, project management can be divided into AI risks and limitations [39]:

- safety: AI technology may not be in line with the security standards followed,

- privacy: AI cannot properly make separation of approved and restricted data and violates the privacy,
- autonomy: as AI dominates the environment, people may feel they are under its control and dependent on it,
- data quality: when data is incomplete or missing, it can diminish the statistical strength of a forecast and result in estimations that may lead to flawed conclusions,
- employment: simple jobs may no longer be available or needed.

Some specific examples of tool assisting project management can be found in literature. A study presents how agile sprint planning could be guided by adapting mathematical MIP (Mixed-Integer Programming) optimization model [40]. Research has shown that estimated time for project deadlines cannot be met due several factors, such as inadequate project scheduling, lack of risk management, lack of change control and undefined project requirements [41]. In this study ChatGPT is considered as a supporting AI tool to create intelligence and help to overcome the pain points found in today’s agile projects. The following chapters describe AI in high level and only the few selected Large Language Models (LLM) implementations in addition to ChatGPT are mentioned.

2.4.1 Generative Artificial Intelligence

The objective of creating intelligent machines is the aim of the interdisciplinary field of science and engineering known as Artificial Intelligence (AI) [42]. Machine learning, a subset of AI and computer science, concentrates on leveraging data and algorithms to replicate human learning, thereby enhancing its precision over time [43]. As typical machine learning (ML) systems require training-set before they can be used in contrast pre-trained ML systems can be used on the shelf.

Natural Language Processing (NLP) is a technology that enables machines to understand human communication by focusing on text analysis, and it's typically used for tasks like text mining, machine translation, and automated question answering [44]. Self-supervised pre-training technique has gained breakthroughs in NLP [45].

Generative AI (GenAI) models harness the power of neural networks to identify patterns within existing data, leading to the creation of original content. These models have seen significant advancements by incorporating diverse learning strategies such as unsupervised or semi-supervised learning in their training process. This has enabled organizations to leverage vast amounts of unlabeled data for constructing foundation models, which serve as the groundwork for AI systems to perform various tasks. [46]

Both Conversational AI (CoAI) and GenAI have become mature enough to provide publicly available tools. The early CoAI system were used by asking a question which the bot would translate as a query that chatbot was trained on. Only a prewritten script and association with the key words was possible. It was not possible to apply context of the topic. Nowadays AI platforms support contextual bias and can understand wide range of language inputs [47]. At time of writing e.g. OpenAI ChatGPT, Google Bard, and Baidu ERNIE are introduced as examples [48]–[50]. These systems are providing Conversational and Generative AI methods for their users.

2.4.2 Large Language Models

According to Shen et al. [51] :"Large Language Models (LLMs) are deep learning models trained to understand and generate natural language". Also, recent studies have demonstrated LLMs achieving success in a wide range of natural language processing as automated summarization of a piece of text, answers to questions based on a piece of text and, translating text from one language to another [51]. Figure 9 shows situation in March 2023 of some of the existing language models size in their parameters [52]. Although GPT-4 has been released it is not officially known what the size of the parameters is used [53].

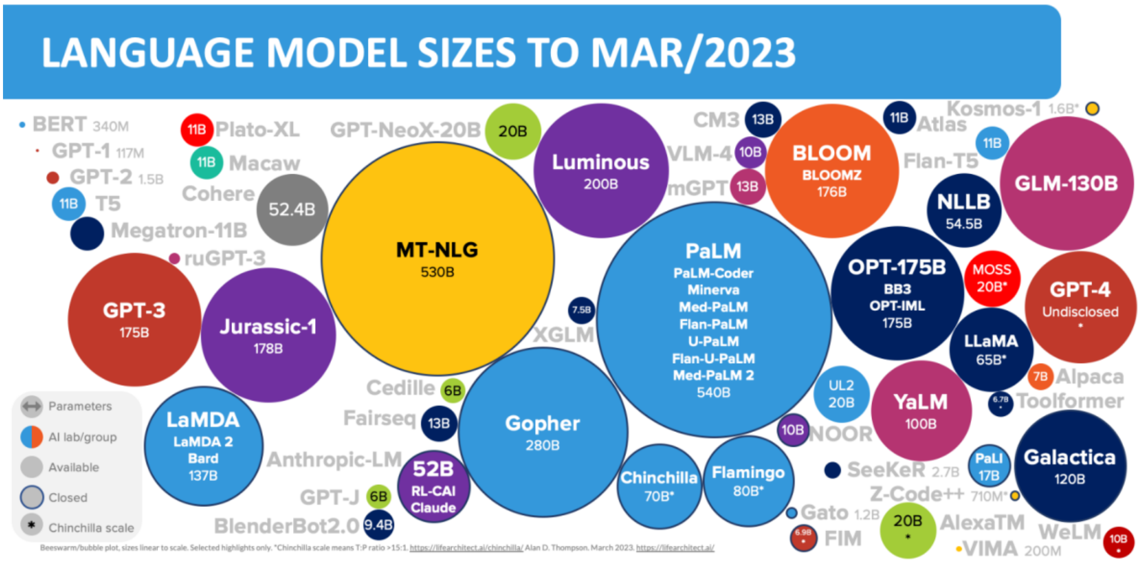


Figure 9. Language model sizes [52]

As LLMs are based on machine learning and before they can work, they need to be trained. The training of a LLM begins by processing a massive volume of data known as a corpus, often reaching petabytes in size. This training involves multiple steps, starting with unsupervised learning using unstructured and unlabeled data, allowing the model to establish relationships between words and concepts. Some LLMs further enhance their training through self-supervised learning, incorporating labeled data to improve concept identification [54]. Recent advancements in transfer learning with language models reveal that comprehensive, unsupervised pre-training is essential for many systems focused on language understanding [55]. Deep learning follows, utilizing the transformer neural network process and its self-attention mechanism, enabling the LLM to comprehend the connections between words and concepts by assigning scores or weights to tokens. Once trained, the LLM becomes a practical tool, capable of generating responses, such as answers, text generation, summarization, or sentiment analysis, when prompted by users. [54]

Transformer architecture has been introduced to support LLMs [56]. The idea of transformer architecture is based on attention mechanism which Vasvani et.al. invented in year 2017 [57]. According to Kerner the different common categories of LLMs include [54]:

1. Zero-shot model: A broad-scale model trained on a general dataset that can provide reasonably accurate results for general use cases without requiring additional training. GPT-3 is often regarded as a zero-shot model.
2. Fine-tuned or domain-specific models: These models undergo additional training on top of a zero-shot model like GPT-3, resulting in a specialized model for specific domains. OpenAI Codex, an LLM designed for programming based on GPT-3, is an example of a fine-tuned model.
3. Language representation model: Language representation models, such as Bidirectional Encoder Representations from Transformers (BERT), utilize deep learning and transformers, making them suitable for natural language processing (NLP) tasks.
4. Multimodal model: Initially, LLMs were primarily optimized for text-based content, but with the introduction of multimodal models, they can handle both text and images. GPT-4 is an example of a multimodal model.

Figure 10 shows the Transformer architecture [57]. The key components of this system are the encoder and the decoder. The encoder is where the text to be processed is introduced into the model. The decoder, on the other hand, produces the processed text, for instance in a different language, assuming the transformer is being used for text translation.

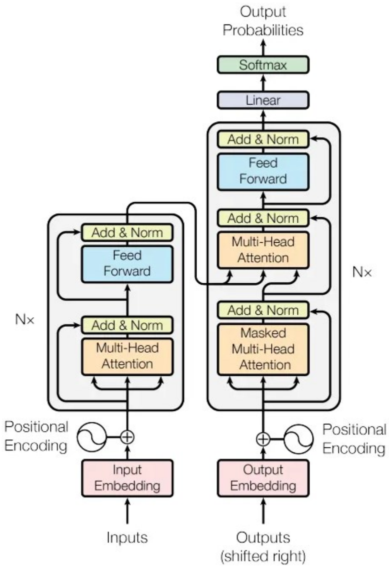


Figure 10. Transformer architecture [57]

The encoder (Figure 10) begins by passing its inputs through a self-attention layer, allowing it to consider other words in the input sentence while encoding a particular word. The decoder, on the other hand, incorporates both layers, with an attention layer positioned between them. This attention layer aids the decoder in concentrating on essential sections of the input sentence. [58]

Several different implementations of the LLMs and Transformer architectures are existing and new implementations are coming in the market [52]. Only few of them are described in the following chapters.

2.4.3 Google Bard

Google calls its Conversational AI (CoAI) as Bard. It combines the knowledge with power, intelligence, and creativity of large language models (LLM). It draws on information from the web to provide fresh, high-quality responses [49]. Bard has been based on Google LaMDA which is Transformer based neural language model specialized for dialog. It has up to 137 billion parameters which are pre-trainee on 1.56 Tera words of public dialog data and web text [59]. The current version of Bard is based on PaLM 2.

Bard explains about itself: “I am a large language model, also known as a conversational AI or chatbot trained to be informative and comprehensive. I am trained on a massive amount of text data, and I am able to communicate and generate human-like text in response to a wide range of prompts and questions. For example, I can provide summaries of factual topics or create stories.” [60]

2.4.4 Bing chat

Microsoft unveiled the latest version of Bing in February 2023, which features an AI-powered web search interface that provides users with summarized search results and a conversational experience. Additionally, users can utilize the platform to create original content such as jokes, letters, and poems. The new Bing runs on an advanced model from OpenAI, which surpasses the capabilities of ChatGPT. Microsoft team has collaborated with OpenAI model to develop a tailored range of abilities and strategies, fusing innovative AI technology with web search. [61]

Bing can respond to users in several different formats, such as traditional links to web content, AI generated summarizations, and chat responses. Web search results will include references and a “Learn more” section with links to search results used as base of the response. It is also possible for users to perform web searches conversationally by adding context to their query and interacting with the system responses to continue further. [61]

2.4.5 Baidu Ernie

Chinese web giant Baidu has relieved to launch own generative AI chatbot during 2023 (Ernie Bot). It is based on Enhanced Representation through Knowledge Integration (Ernie). It is being said: “What sets ERNIE apart from other language models is its ability to integrate extensive knowledge with massive data, resulting in exceptional understanding and generation capabilities. “ [50]. PCL-BAIDU Wenxin a.k.a. ERNIE 3.0 Titan pre-trained language model has 260 billion parameters which has been trained on massive unstructured data a gigantic knowledge graph and having both natural language understanding (NLU) and natural language generation (NGU) [45]. The ERNIE framework can be seen in Figure 11.

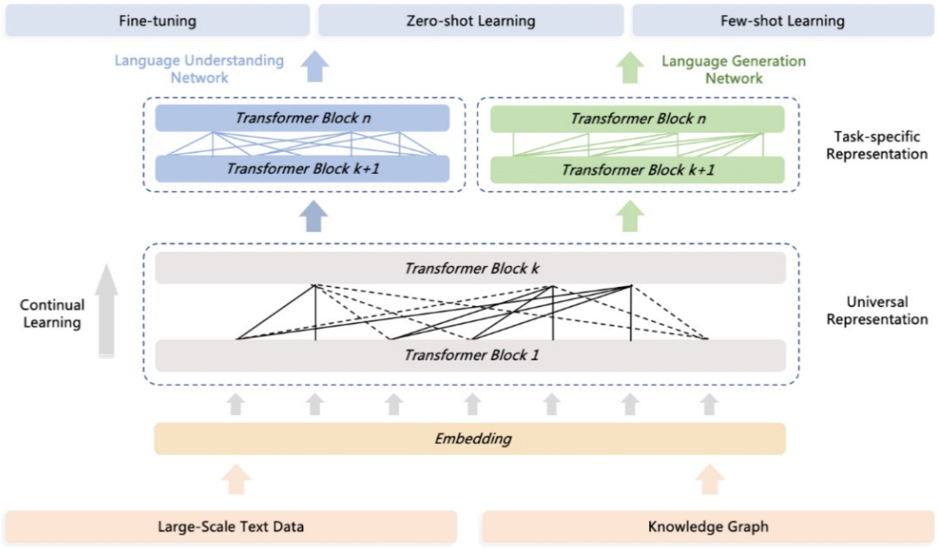


Figure 11. The framework of ERNIE 3.0 Titan [45]

According to Baidu Research, ERNIE promotes controllable learning and creditable algorithms. The controllable learning algorithm enables the model to compose in a targeted, controllable manner, with given genre, sentiment, length, topic, and keywords [45]. The credible learning algorithm trains the model to distinguish fake, synthesized language from real-world human language through a self-supervised adversarial learning framework [45].

2.5 ChatGPT

ChatGPT is Large Language Model (LLM) chatbot that has been developed by OpenAI on top of GPT-3.5 and GPT-4 (March 2023). It uses ML algorithms to generate text based on the input provided by a user. It is notable the GPT-3.5 model of ChatGPT can reference up to approximately three thousand words (or four thousand tokens) from the current conversation and any information beyond that is not stored [62]. Moreover, ChatGPT is not able to access past conversations to inform its responses [62].

It is used through web address that provides conversational user interface. As its name GPT refers to Generic Pre-trained Transformer [53]. ChatGPT is conversational chatbot AI and it is based on Large Language Model and is publicly available for registered users [63].

2.5.1 Training data

The model behind ChatGPT was trained on a large corpus of text data, allowing it to generate human-like responses to a wide range of questions and prompts [48], [63]. It has an additional layer used in training called Reinforcement Learning with Human Feedback (RLHF) which improves ChatGPT ability to follow directions and generate responses in human understandable format [64]. RLHF has emerged as a key method to finetune LLMs and align them with human values [42]. This involves humans ranking language model outputs sampled for a given input, using these rankings to learn a reward model of human preferences, and then using this as a reward signal to finetune the language model with using RLHF [42].

According to OpenAI an initial model was trained through supervised fine-tuning, where human AI trainers carried out conversations while playing both the user and the AI assistant. Trainers used model-written suggestions to help compose their responses. For reinforcement learning, a reward model was needed, which involved collecting comparison data: rankings of quality for two or more model responses. This data was collected from conversations between AI trainers and the chatbot. A model-written message was chosen randomly, several alternative completions were sampled, and AI trainers ranked them. A specific Proximal Policy Optimization was then used to fine-tune the model based on these reward models. This process was iterated several times.[65]

2.5.2 GPT-4

GPT-4 is a large multimodal model capable of processing image and text inputs to generate text outputs [53]. The model has potential applications in dialogue systems, text summarization, and machine translation, among others. It aims to enhance understanding and generation of natural language text, especially in complex scenarios [65].

GPT-4 performs well on a range of exams intended for humans, often outperforming most human participants [53]. According to OpenAI GPT-4 significantly reduces hallucinations relative to previous GPT-3.5 models [53]. At the moment the use of GPT-4 features requires an additional monthly paid service fee and is thus not freely available for everyone.

2.5.3 Limitations and Challenges

ChatGPT can utilize GPT-4 text input capability which has been introduced in March 2023. According to technical reports: “Despite its capabilities, GPT-4 has similar limitations to earlier GPT models [1, 31, 32]: it is not fully reliable (i.e., can suffer from “hallucinations”), has a limited context window, and does not learn from experience. Care should be taken when using the outputs of GPT-4, particularly in contexts where reliability is important.” [53]. According to MIT review: “GPT-4 can still generate biased, false, and hateful text; it can also still be hacked to bypass its guardrails.” [66].

Based on these reviews GPT-4 language model should be used with care in mind. According to article LLM models generate the most fitting responses based on the vast volumes of internet data they've been trained on [67]. However, they lack the ability to discern between factual and inaccurate information [67].

2.5.4 Applications and Use Cases

GPT models and generative AI are being employed by various organizations in different sectors for tasks such as creating social media content, converting text to different styles, writing, and learning coding, analyzing data, and Q&A bots, generating text summaries, producing content, producing learning materials, building interactive voice assistants, and enhancing search capabilities (see chapter 2.4.4; Bing chat) [68]. Furthermore, one of the interesting use cases is assisting in agile project management and its pain points.

As ChatGPT provides generic LLM it can be used for various expert systems to provide response to any given prompt. Development for different plug-ins benefiting of GPT features has started and at the moment different tools for e.g., software development is provided such as GitHub copilot [69].

3. RESEARCH MODEL: THE AGILE PAINT POINT MODEL

Agile projects may contain a range of challenges, particularly as the complexity, context, and scale of agility changes. Numerous research documents have identified different challenges [70][71][72][73][74][75][76][77][78][79]. To better manage these challenges, a specific pain point model has been constructed to analyze some of the most common pain points encountered in literature related to agile projects. The aim of this model is to provide an explanation for the typical reasons behind these challenges and propose practical solutions for overcoming them independently on the size of the project.

To create this model, first a classification of different challenges was made based on a literature search using key words related to agile project challenges, with a focus on papers published the last ten years and not focusing any certain agile methodology or scale of project in mind.

3.1 Agile pain points in literature

Literature research goes through different publications and tries to pick the most commons challenges found in agile projects. Starting from Patel et al. [73] found in research that members of the development team having a background using structured methodologies like the waterfall approach may resist the adoption of agile practices. Furthermore, other challenges mentioned include a lack of understanding of the agile methodology by the development team, insufficient involvement of top management, technical support issues, incompatibility with existing infrastructure and processes, and the presence of budget and time constraints [73].

Research by Nuottila et al. conducted on public sector agile projects and identified several areas where uncertainties can arise. These areas included documentation, education, experience, and commitment, stakeholder communication and involvement, roles within the agile set-up, location of agile teams, legislation, and complexity of software architecture and systems integration. [70]

Furthermore, it has been studied that different agile methodology is mixed such as Scrum, XP, and Lean [71]. In this manner agile methodology is defined in quite widely scope. Challenges faced by agile practitioners are complex and context-dependent, with some areas of difficulty having persisted for many years, making them hard to successfully address. Conversely, while some challenge areas persist, their focus has shifted. While practitioners have adopted agile methodologies, concerns regarding their continued effectiveness have arisen. Some challenge areas, such as governance, business engagement and transformation, failure, and the impact of claims and limitations, have not received widespread research attention. Moreover, certain challenges, including governance and contracts, remain under-researched, while others such as business and IT transformation have been researched but have failed to have the expected impact on industry practice. [72]

Research by Dikert et al. continues to list challenges in scaling agile such as [71]:

- change resistance at different organization level,
- difficulties to implement agile due misunderstand concepts,
- lack of coaching and training,
- high workload,
- old commitments,
- physical availability of people,
- endurance by maintaining agile way of working,
- different teams interpret agile differently,
- requirements management and refinement,
- work estimation for user stories.

Also, selecting large-scale agile framework is not easy. According to Conboy et al. different agile frameworks are understandable at the basic level, but when they are applied outside of their intended context of a specific framework, guidance quickly runs out [75].

Starting from year 2019 due to Covid-19 pandemic has had a different effect in the entire world causing remote project work increasing [80]. Remote project work has introduced own challenges. Study on this subject has discovered challenges such as [74]:

- fewer organic interaction opportunities,
- lower engagement,
- meeting overload,
- leaders may take more control and not supporting,
- less interaction reflects in less knowledge sharing.

Different solutions are presented to the above challenges. According to the research by Reunamäki et al. creating smaller sub-teams (2-4 members) within the existing teams solve the problem of closer interaction. As a drawback sub-teams tended to produce informal and invisible hierarchies. Team members were also encouraged to take turns in organizing different sub-team and larger team's meetings and ceremonies. Having cameras on helped to increase engagement and presence of team members. The danger was that prioritization of other meetings caused inflation in meetings by lowering the participation activity. In addition, remote agile causes meeting overload as number of online meetings increase and experts started to reserve own time in their own calendar for non-meetings. It was shown that leaders may take more control and become more micro-managing and not provide enough support when working remotely agile. To overcome this more presence and interaction with employees were promoted. Less integration within the organization level occurred as large auditorium could not be used in-person meetings. This could be help with the assistance of the software solutions supporting information sharing meant for agile work. At the end it is useful to consider if agile is likely to work equally in different conditions.[74]

Challenges found when transiting to agile organisation has been opened in Paasivaara et al. research on global company taking agile methods in use: Change resistance, technical debt, lack of a common agile framework, lack of coaching and coaches, lack of agile training, cross-site teams, working as “a real team”, any team cannot implement any feature, lack of continuous integration and test automation, agile teams in a waterfall organisation, challenges in defining the product owner (PO) role, challenges in breaking down the requirements, backlog challenges, and constant change. [81]

Figure 12 presents division of agile project management challenges into four categories according to research by Hoda et al. related to self-organizing teams [76]. These categories describe the challenges in the project, team, individual and task levels. The project level contains activities related to involving the team, SM, and customers; team level included activities involving only the core development team and their SM; individual level had activities of team members and task level the technical tasks. [76]

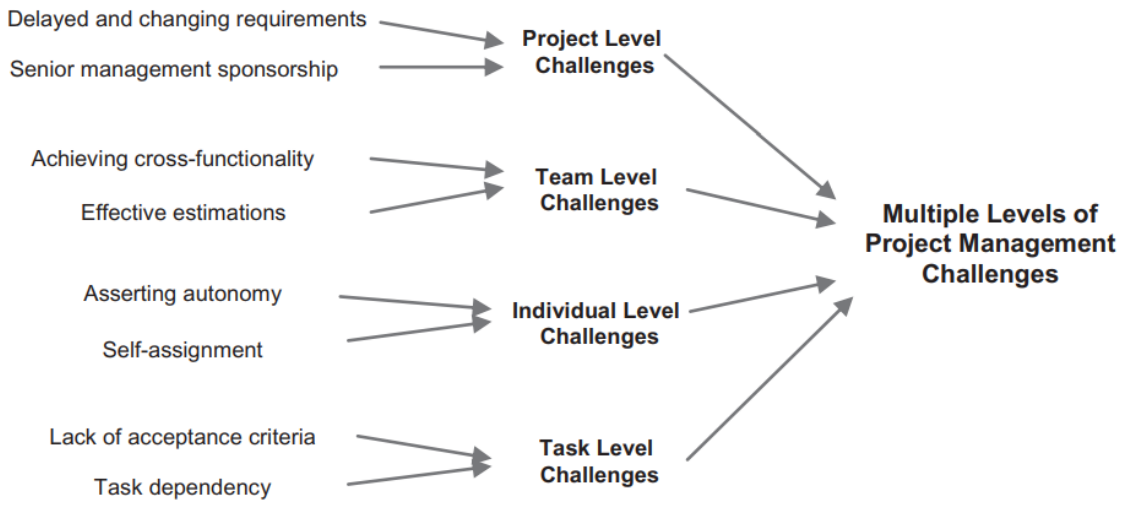

Delayed and changing requirements
Senior management sponsorship
Project Level Challenges
Achieving cross-functionality
Effective estimations
Team Level Challenges
Asserting autonomy
Self-assignment
Individual Level Challenges
Lack of acceptance criteria
Task dependency
Task Level Challenges
Multiple Levels of Project Management Challenges

Figure 12. Agile project challenges [76]

In the same study conducted by Hoda et al., challenges at the project level were found to be related to [76]:

- Delayed and changing requirements, as understanding customer requirements is crucial for developing effective software.
- Senior management sponsorship, as it can be difficult for project teams (including project managers and team members) to convince senior management to adopt agile methods and embrace a self-organizing context.

Furthermore, team-level challenges were identified as [76]:

- Achieving ideal cross-functionality, as this requires team members with varying specializations, thought processes, and behaviour patterns to interact and better understand each other's perspectives, thus minimizing personnel dependency.
- Making effective estimations, as arriving at a unanimous estimate can be challenging, especially considering the time constraints involved, and experienced team members tend to buffer extra time while newer members may underestimate the workload.

Individual challenges were also noted in the study [76]:

- Asserting autonomy, which varied among team members, with some being self-motivated and taking responsibility for their own duties.
- Self-assignment, which indirectly threatened cross-functionality in the team as members tended to pick tasks in their comfort zone, leading to increased specialization and a loss of cross-functionality.

Moreover, task-level challenges were also identified in the study [76]:

- Lack of acceptance criteria when customers did not provide clear acceptance criteria, requiring agile teams to seek clarifications and acceptance criteria and not just elicit requirements from the customer.
- Task dependency between tasks within a sprint or across sprints, which could lead to cancelled sprints and decreased project performance and velocity.

Sithambaram et al. divided issues and challenges into organizational, people, process, and technical factors in their grounded theory (GT) approach for agile-hybrid projects [79]. GT is a research methodology that aims to develop theories based on empirical evidence derived from real-world situations. At the organizational category factors are related to lack of executive sponsorship, lack of management commitment, organizational culture not supporting the change, misunderstanding of agile methods value, organizational objectives, and agile principles not met, and organizational politics [79]. People factors contains lack of agile skillset, project management incompetence, lack of teamwork, poor customer relationship, lack of proper communication culture, stakeholders do not embrace the change, lacking agile mindset and understanding, and team empowerment [79].

According to Sithambaram et al. process factors encompass a different set of issues such as poor project scope management, inadequate management of requirements, insufficient project planning including reporting and tracking, undefined customer roles, minimal project governance, improper implementation of agile methods, ineffective cost management, unchangeable processes, and the use of agile or traditional methods without considering the project's context. Other process-related factors include the lack of well-defined project manager roles, poor prioritization and scheduling, and inadequate handling of project closures such as failing to maintain lesson learned or conduct retrospectives. On the other hand, technical factors are characterized by the availability of technology, knowledge, and tools, as noted by. [79]

According to Shameem et al. challenges can be classified into management, team, technology, and process in distributed software development (DSD). The distributed software development paradigm is a contemporary approach utilized for developing commercial software that meets the needs of a global client organization. One of the examples of these challenges is coordination between the teams. [77]

Furthermore, the challenges, factors affecting effort estimation has been studied [78]:

- organisation: organisational alignment, executive support, organisational politics,
- people: team stability, skills and knowledge, team familiarity, commitment, and communication,
- process: requirement refinement, agile maturity, regular delivery, work in progress, user involvement,
- technical: dependencies, poor code documentation, unreliable infrastructure, bugs or incidents, lack of code quality, insufficient testing,
- project: size, newness, security, communication

As conclusion different and versatile challenges can be found in agile projects. Albeit challenges may be classified and combined to create pain point model for the most common challenges found in agile projects. Agile project management contains lots of different literature and experiences in taking agile project management way of working in use. Many paint points reflect the challenge in paradigm shift from traditional (a.k.a. descriptive) project management to agile project. However, pain points may be similar to general project management and are not necessarily related to agile way of working.

3.2 Construction of pain point model

The prior chapter outlined various challenges encountered in contemporary agile projects as identified in the research literature. A distinct model addressing these pain points is introduced, aiming to provide solutions for common issues in agile endeavors. To devise this model, challenges were categorized. Although numerous classifications exist on the subject, this study proposes one potential arrangement, acknowledging that some challenges might span multiple categories. Five distinct categories were identified, and within each, two predominant challenges were chosen based on their prevalence in academic literature. These documented challenges then informed the suggested solutions to these prominent pain points. To categorize challenges pertaining to pain points, analogous studies on agile projects were analyzed, with their results displayed in Table 1. The classification system somewhat mirrors the one by Sithambaram et al. [79], which includes categories like project, people, process, organizational, and technical. However, this study replaces "organizational" and "technical" with "endurance" and "effort estimation." In this context, "endurance" predominantly alludes to resistance to change, and the sustained commitment to adhering to agile principles and practices.

Table 1. Classification of agile pain points.

Classification	Description	References
Project	Factors both within and outside the project, such as managing requirements, stakeholder and management support, and communication, influence the project's execution and its ability to meet its needs.	Nuottila et al. [70] Dikert et al. [71] Hoda et al. [76] Vlaanderen et al. [82] Kamal et al. [83] Sithambaram et al. [79] Kula et al. [78]
People	Issues related to individuals, including skill deficits, clarity in role delineation, communication practices, understanding of agile principles, and empowerment.	Nuottila et al. [70] Dikert et al. [71] Gregory et al. [72] Patel et al. [73] Reunamäki et al. [74] Sithambaram et al. [79] Moe [84] Hoda et al. [76] Barke et al. [85]
Process	A broad comprehension of the significance of agile processes and the flexibility to adapt these processes.	Gregory et al. [72] Patel et al. [73] Kula et al. [78] Barke et al. [85]
Endurance	Sustaining an agile work approach without reverting to previous practices and confronting resistance to organizational change.	Dikert et al. [71] Gregory et al. [72] Sithambaram et al. [79]
Effort estimation	Estimating the workload can be challenging, and executing the	Dikert et al. [71] Hoda et al. [76]

	required tasks may be hindered due to insufficient technical expertise, inappropriate tools, and a lack of knowledge.	Kula et al. [78] Sithambaram et al. [79]

Picked from Table 1, a fishbone diagram illustrating the major pain points can be drawn (as seen in Figure 13). This diagram displays two distinct challenges for each problem category identified through the literature review.

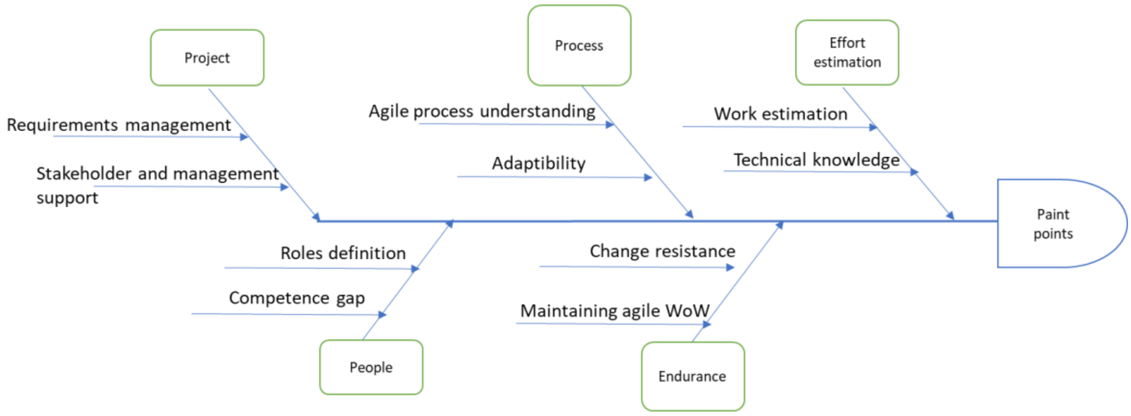


Figure 13. Paint points

In the subsequent chapters, possible solutions are proposed for overcoming each identified pain point.

3.2.1 Requirements management

Kamal et al. have investigated Agile Requirement Change Management (ARCM) and identified best practices from the literature [83]. Although the context has been Global Software Development (GSD) the two most favorable findings were effective communication and requirement traceability [83]. These findings can be seen corresponding agile way of working, individuals and interactions and responding to change [86]. Laanderen et al. have focused on Agile Requirement Management and proposed an Agile Software Product Management (SPM) approach for handling requirements [82]. For large software development teams, a steady flow of well-defined and approved requirements is crucial, and a functional team of product managers is needed to work closely with the development team [82]. Agile project management tools already support requirements management. One example of these tools is Jira [87].

According to PMI agile practice guide project sponsors and stakeholders can be helped to craft a product vision [3]. Moreover, bringing the team and product owner together can meet the expectations [3]. And lastly, decomposing roadmap into backlog of smaller and concrete requirements [3].

3.2.2 Stakeholder and management support

Senior management may hesitate to experiment with new methodologies, while changed requirements may occur at any time during the project's lifecycle, disrupting the normal sprint cycle [76]. If management is committed to supporting executive sponsorship with a clear understanding of the values and principles of agile projects, along with well-defined business objectives and a culture that aligns with these values, then success is likely to follow [79]. Executive sponsors can play a vital role in ensuring the success of projects and helping companies achieve their strategic initiatives and goals [88].

It is important to understand the various roles that executive sponsors can, will, and should play to maximize their positive impact [88]. Therefore, it would be beneficial to keep the project steering group and various stakeholders informed of the benefits of agility to gain the maximum support. In addition, according to study executive management support has major responsibility for the success and the failure of the project [89]. Agile practice guide from PMI recommends servant leadership to effectively work with stakeholders who make impossible demands [3].

3.2.3 Role definition

An agile team may lack of cross-functional capabilities. Team is composed based on functional expertise and not leadership abilities [90]. Also, same people are targeted which results in overburdened team members [90]. Lack of redundancy and highly specialized skills can be barrier for self-organizing team [84]. Unclear roles in agile teams have challenges as studied [85]:

- When there is a lack of comprehension regarding team roles, self-organization can prove to be challenging,
- it’s a common occurrence for this to happen unexpectedly,
- reflecting on roles with the assistance of a coach can potentially alleviate this issue,

- however, achieving a sufficient level of understanding and acceptance is still no easy feat,
- once attained, a strong comprehension and acceptance of roles can significantly enhance a team's effectiveness and collaborative efforts,
- the process of obtaining role clarity is demanding and may present emotional difficulties.

Based on the finding above clear role definition would be beneficial in team work to clarify responsibilities. This would help in self-organization as team members could concentrate on certain tasks. However, if same team members having clear roles are overburdened all the time the only way to overcome the situation might be to add more similar roles and skillsets. It is good to remember also Scrum Master role who could be the ideal person to overcome blocking teamwork when roles are not clear. Scrum master is the role who is responsible to remove the blocking issues in the team [18].

3.2.4 Redundancy

Required competences have similarities with team roles definition. Gregory et al. (2014) argues that successful implementation of agile requires skilled, self-directed, and motivated team players, as well as effective team practices [72]. However, individual goals may sometimes take precedence over team goals, and the highly specialized skills of team members may lead to a lack of redundancy [84].

To create an innovative project team, four levers can be used: diverse and targeted team composition, talent management, iterative goal setting, and continuous learning [90]. Moreover, being self-reflective may be more important than technical proficiency [90]. Based on the previous findings it seems that overcoming challenges in competence gap requires good talent management and continuous learning that different skillset can be developed as required in the teamwork.

3.2.5 Agile process understanding

Understanding agile processes is one of the important challenges. Organizations that value agility develop a strong foundation of key processes that mitigate potential obstacles to success and enable the organization to utilize new opportunities [88]. These processes help maintain consistency in project delivery and enable more effective monitoring and measurement of success [88]. Furthermore, they serve as a solid foundation for future process improvement initiatives as needs arise [88].

Three process drivers are recommended that organizations with high agility should use to balance speed and stability [88]:

- Comprehensive sets of project management tools that encompass predictive, Agile, and hybrid methodologies.
- Professionals with a wide range of technical skills and soft skills that enable them to excel as leaders and coaches, think creatively and innovatively, build effective teams, and expertly manage relationships - particularly with customers.
- Efficient processes for managing change and risk that minimize the impact of external forces and leverage any potential opportunities they may bring.

According to Boem et al. the project's needs should be evaluated and only those processes should be selected that seem indispensable and build up instead of tailoring them down [91]. Been said, efficient processes play vital role in agile projects and help organizations to measure and improve their way of working. Effort should be spent to get them right and fit for the purpose for the organization.

3.2.6 Adaptability

Organizations that perform at a high level possess the ability to adapt, adjust, and innovate on an ongoing basis [88]. Achieving this capability requires a conscious effort to engage in experimentation and learning [88]. Barke et al. continues highlighting coaching in self-organization: “Agile coaches are usually considered facilitators for a team's self-organization and indeed this is what we find.” [85].

For agile teams, the ability to adjust to various shifts is crucial for ongoing enhancement. Leveraging the expertise of an agile coach can be an invaluable strategy, providing teams with the guidance and tools needed to navigate these changes effectively, thereby ensuring their resilience and continued growth in an ever-evolving environment.

3.2.7 Change resistance

Change resistance and maintaining agile is difficult without reverting to old habits. Study has shown that this can happen when the combination of schedule pressure and lot of changes at once can pull people back to the old way of working [71]. Furthermore, for agile to become a lasting and integrated part of teams and the organization, continual change and dedication are necessary [72].

According to Patel et al. scrum masters who possess strong group management skills and empathy understand the importance of facilitating team collaboration by selecting the right members to participate in the team [73]. In addition, they can establish trust among team members who may hold differing views by utilizing reasoning, compromise, and open discussions to explore and evaluate legitimate options [73].

3.2.8 Maintaining agile way of working

Maintaining agile way of working without restoring back to the old behavior can be difficult. According to Dikert et al. one of the challenges often encountered is the difficulty of implementing agile methodologies [71]. Although a software team may excel in training, they may struggle when it comes to applying agile techniques in practical situations, potentially leading to confusion and setbacks [71]. Also, project management expertise has been found the most important as control of continuous interaction and feedback with the project team to quickly make decisions [89]. Furthermore, study shows that project management incompetence or lack of the necessary skills set to effectively apply agile principles may hinder expectations of different stakeholders [79].

To sustain an agile work approach, it's imperative that the management not only possesses robust project management skills but also a deep comprehension of agile methodologies and their underlying principles. This ensures that the team can navigate challenges effectively while staying true to agile tenets.

3.2.9 Work estimation

One of the difficult tasks in agile project is an estimation of a story point or generally how much effort is needed for certain task. Story points are units of measure that are used to estimate of the overall effort required to fully implement an individual product backlog item. Studies show that creating user stories is hard [71]. Also, learning is needed from product management and development [71]. According to Usman et al. variety of estimation techniques have been applied in an ASD context but those used the most are the techniques based on some form of expert-based subjective assessment [92]. Different methods are used to estimate a story inside of the team e.g., Planning Poker, Analogy, and judgement by expert and considering the amount of work, complexity, risk, and uncertainty as Fu and Tantithamthavorn explain [93]. They have researched a GPT-

2 pre-trained language model with a GPT-2 Transformer-based architecture and produced GPT2SP models to better capture the relationship among words while considering the context surrounding a given word and its position in the sequence and be transferable to other projects, while being interpretable [93].

As conclusion in their research, they present that in addition, 69% of the respondents consider adopting AI-based Agile Story Point Estimations if they are integrated into modern software development tools (e.g., JIRA), highlighting the practical need of our Explainable AI-based story point estimation approach [93]. At the end of the day, it is recommended to reduce story size by splitting them and use relative estimation having the whole team participating for more accurate effort estimation [3].

3.2.10 Technical knowledge

Lack of technical knowledge and suitable skillset play vital role in agile project teams. According to colonies in small projects, the success lies on the shoulders of the individuals who make up the team, making it important to show greater respect towards skilled resources as they play a crucial role in the project's outcome [89]. Inappropriate, or insufficient technology and tools may cause a challenge in effort estimation. According to Sithabaram this can happen e.g., when a complex bug is found and no tools nor knowledge of using tools may end up taking a considerable amount of time to Figure out the actual problem and cause manual work instead of utilizing tools and technologies for fixing the problem [79]. When complex bug fixing takes time it may cause interruption to the teamwork when almost everyone is on top of the problem. In worst case the project may be delayed due critical bug fixing. Furthermore, tasks performed might be too challenging and take more time after all [94].

Technical expertise is pivotal for accurate effort estimation and necessitates regular oversight throughout the project's execution. To address this, it's essential to align each individual's capabilities with corresponding project tasks. By ensuring that the right expertise is paired with the appropriate tasks, the likelihood of project completion according to the plan increases. In this context, employing a SWOT (Strengths-Weaknesses-Opportunities-Threats) analysis to evaluate the necessary skills can be a beneficial approach.

4. RESEARCH METHOD: DESIGN SCIENCE RESEARCH

In this chapter, the research process and methodology get delineated as shown in Figure 14. The initial phase of the study introduces various agile frameworks. Illustrations of these frameworks in larger, scaled-up applications in substantial agile projects make up part of this exploration.

These frameworks receive classifications into small-scale and large-scale. The small-scale group encapsulates Scrum, XP, Kanban, and Lean Software Development. On the other hand, the large-scale frameworks include SAFe, LeSS, and DA. The selection of these frameworks derives from insights culled from pertinent literature. The goal remains to provide an encompassing introduction and guidance on these frameworks' use.

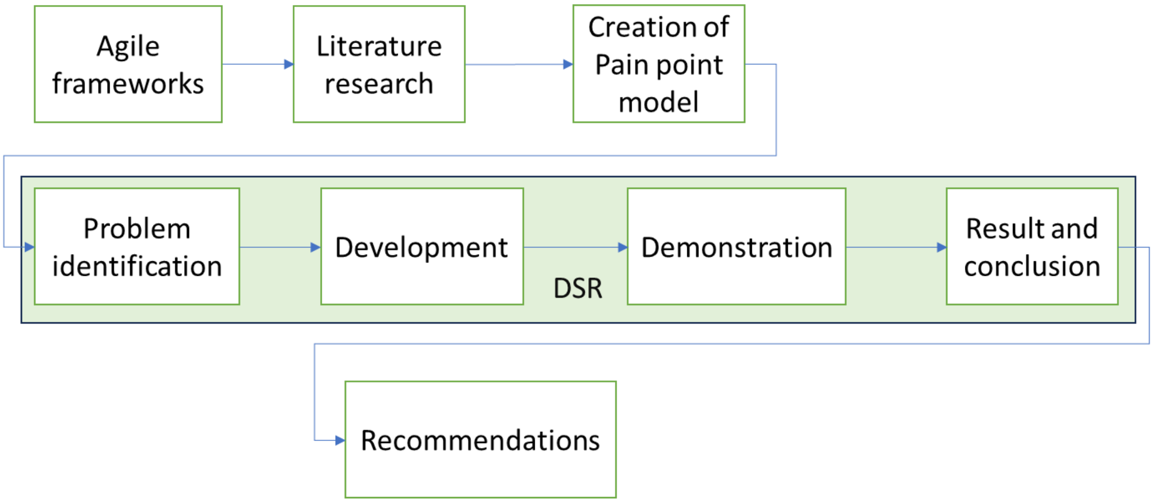


Figure 14. Research process

Moving on to the next section, it involves an extensive literature review on the common challenges encountered when adopting and implementing agile methodologies. These challenges are analyzed, categorized, and synthesized into a pain point model, which is presented as the problem setting for the research.

Design Science Research (DSR) has been employed as methodology for a strategic approach to discover effective AI solutions for mitigating these identified pain points (Figure 14). DSR is an approach to problem-solving that aims to advance human knowledge through the development of innovative artifacts [95]. These artifacts, called prompt patterns in this study, are designed to address specific challenges, and enhance their surrounding environment, result in an enriched technology and science knowledge base [95]. In DSR research is conducted first identifying the problem, defining the objectives, developing the solution, demonstrating, and evaluating the results [96]. Finally, practical recommendations are made.

4.1 Problem Identification and Objectives Definition

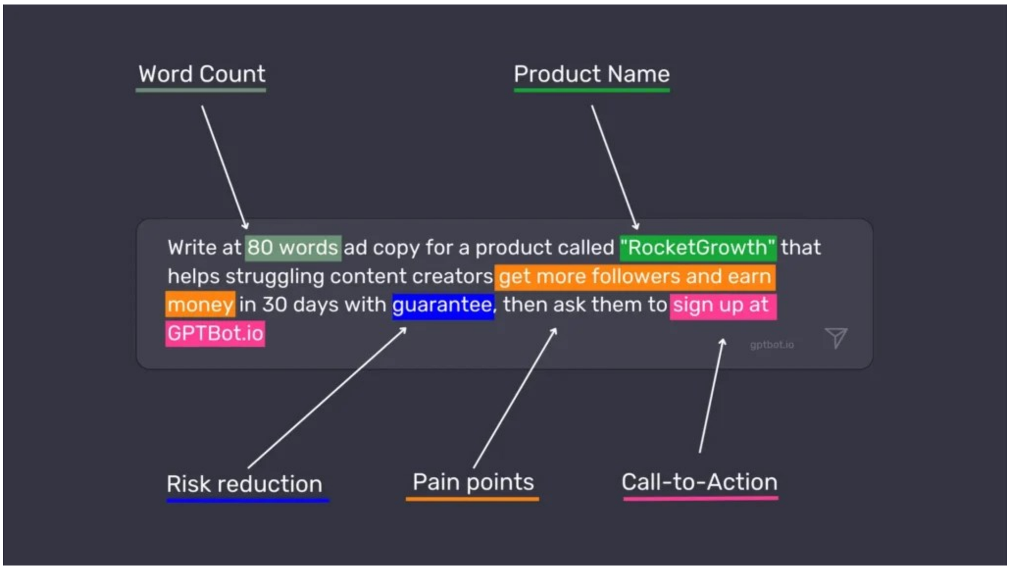
The identified problem is the formulation of appropriate prompts to be used in conjunction with ChatGPT, aimed at easing the paint points commonly associated with agile projects. Primary objective to assess the possible implementation of ChatGPT as a support mechanism in intelligent agile project management. Moreover, objective is to determine distinct prompt patterns that generate precise information. While the creation of prompts can take on many forms, this research does not develop a specific grammar, but instead designs patterns to steer ChatGPT toward providing suitable responses with minimal hallucination, a method supported by White et al. [97]. The prompt patterns used in this study adopt a similar strategy, abstaining from introducing a unique syntax or language. The aim is to supply relevant keywords that can aid project managers or stakeholders in initiating early dialogues with AI, thereby broadening the application of project-specific parameters.

The problem identification in this research hinges on the empirical aspect of design science research, which involves interacting with ChatGPT to evaluate various prompts capable of generating accurate responses for a specific subject, aligning with the method proposed in the White et al. studies [97]. Unlike focusing on prompt patterns to improve code quality, the emphasis is on identifying and assessing prompts that can support agile projects while mitigating the impact of various challenges.

4.2 Development

In the development phase of the DSR method, prompt patterns (artifacts) are generated for ChatGPT, designed to assist in mitigating the challenges associated with agile methodologies. A prompt, as defined, is a textual input given by the user, acting as the commencement point for ChatGPT's response generation [98]. A prompt pattern, therefore, is a generalized construct for a specific prompt topic.

The development of these prompt patterns involves using ChatGPT's web-based interface along with the GPT-4 model. The intention behind this phase of the research is to create a practical and robust means of addressing agile project pain points through specifically crafted prompts.



Write at 80 words ad copy for a product called "RocketGrowth" that helps struggling content creators get more followers and earn money in 30 days with guarantee, then ask them to sign up at GPTBot.io

Figure 15. Example of prompt [98].

According to White et al. a prompt (Figure 15) sets the context for the conversation and tells ChatGPT what to focus and what are the expectations for the output [97]. A specific prompt pattern is implemented to each specified pain point. In conversations with AI different types of prompts: explicit, implicit, and creative can be used. Explicit prompts are direct and clear instructions given to the AI model about the specific format or information needed in the output. On the other hand, implicit prompts are less direct and give the AI model more flexibility to interpret the intended result. Creative prompts aim to inspire AI models to produce original, imaginative, or unconventional outputs. [99]

Explicit approach for prompt pattern development has been selected. Each prompt pattern developed follows roughly the model introduced by White et al. [97]:

- The name and classification. The name is used for unique identify the prompt pattern. The classification is based on the presented pain point model classification defined in chapter 3.2.
- The intent. The purpose of the pattern is conveyed through its intent.
- The motivation behind the pattern is documented, which explains the underlying problem it is intended to solve and why it is important.

- The structure and key ideas. The structure and key ideas describe the fundamental contextual information that the prompt pattern provides to the ChatGPT.
- The demonstration. Providing example pattern helps to demonstrate how the pattern can be applied in agile projects.
- Results are presented as empirical observations and summarizes the advantages and disadvantages of implementing the pattern in practical situations.

Prompts should set the context, define expectations, channel creativity, and reduce ambiguity [98]. Each prompt pattern developed contains the following contextual sentences:

1. ChatGPT (LLM) is asked to act as [roleA] to work (task) in given problem domain.
2. The necessary constraints are presented [n * C].
3. ChatGPT (LLM) is asked to verify understanding and clarifications of the constraints.
4. ChatGPT (LLM) is asked to reflect its understanding and providing solution to [roleB].
5. Finally, the output format is requested, defined as [format].

Both roleA and roleB represent different and typical project staff roles such as project manager, engineering manager, program director, software developer, and requirements engineer. Constraints can be given as free description or comma separated items. Constraints (n * C) can vary from requirements to different objectives according to project needs and are subject to each project.

4.3 Demonstration and Evaluation

The prompt pattern's efficacy is assessed through practical demonstrations. Each prompt is entered using ChatGPT and response is collected for evaluation. Evaluation is done for each prompt and summary is presented as contribution. Since the outcomes might be subjective and immeasurable, reference to existing literature is employed to evaluate the effectiveness of each prompt pattern. In demonstrations, hypothetical project management challenges are utilized. Every demonstration of prompt patterns occurs three times, utilizing the same prompt, ensuring consistency in responses from ChatGPT. During the third issuance of prompts, an additional iteration ensures further consistency.

The research discusses theoretical and practical implications derived from literature findings and observations, offering practical recommendations on how ChatGPT can be employed in agile projects pain points. Ultimately, the research aims to tackle the proposed research questions.

5. RESULTS

This chapter showcases various prompt patterns and corresponding demonstrations utilized with ChatGPT. Given that ChatGPT can generate extensive responses, only selected portions of these dialogues will be highlighted in the subsequent chapters. Complete, original responses can be referenced in the appendices. The result section exhibits each prompt pattern via a sample dialogue with ChatGPT. Each prompt is inputted into ChatGPT thrice, spanning three rounds within a single atomic session, to observe the variations in ChatGPT's responses to identical prompts. The displayed prompt examples are selected mostly from the first round, and the full dialogues and results from other rounds are available in Appendices A, B, and C. The last prompt test round contains additional prompt iterations.

Prompt patterns are classified according to Table 2 so that there are two patterns representing classified pain points. As of June 2023, ChatGPT operates with a maximum token limit of 2048 for a single prompt [100]. However, the demonstrations employ a specific tool that utilizes a smaller token size verified by tool [101]. Testing has revealed that if the prompt size exceeds this limit, it hampers ChatGPT's ability to respond. It might even cause the model to forget previously discussed context [62]. Nevertheless, during prompt demonstrations, such behavior was not encountered.

Table 2. Prompt pattern classification.

Classification	**Pain point**	**Prompt pattern name**
Project	Requirements management	Requirements creation pattern
	Stakeholder and management support	Steering group pattern
People	Role definition	Role clarification pattern
	Redundancy	Redundancy analysis pattern
Process	Agile process understanding	Agile process coaching pattern

	Adaptability	Adaptability management pattern
Endurance	Change resistance	Change resistance pattern
	Maintaining agile way of working	Agile way of working pattern
Effort estimation	Work estimation	Work estimation pattern
	Technical knowledge	Technical knowledge pattern

5.1.1 Requirements management

Assessing the utility of ChatGPT in assisting project requirements generation, the intention behind this prompt pattern has been to aid in formulating accurate needs that will enable an agile project to address the pain point associated with requirements management and creating requirements. The motivation for this prompt structure has been to assist in establishing well-defined, comprehensible requirements while also facilitating their concretization and traceability. Using this pattern (Table 3) would help project to create initial requirements and furthermore generally in requirements management.

Table 3. Requirements creation pattern

Contextual statements

`As a [roleA] your task is to create a requirement specification.`
`The requirement specification must fulfil the following constraints: [c1, c2, c3…].`
`Ask the necessary questions to fully understand the constraints.`
`Based on my answers create the requirement specification.`
`Explain your understanding and construct your proposal to me representing the [roleB].`
`Provide output in [format].`

Prompt pattern demonstration for ChatGPT: “As a requirement engineer your task is to create a requirement specification. The requirement specification must fulfill the following constraints: Internet banking system which can handle typical banking transactions to be possible to browse account and make payments. Ask the necessary questions to understand the constraints. Based on the answers create requirement specification.

Explain your understanding and construct your proposal to me representing the project manager. Provide your output in Table format.”

For this prompt ChatGPT response in the first prompt test round is*: “Sure, I will need a few more details to make sure we're on the same page. Here are the questions that I'd need answers to…”*

EC1: During each iteration, ChatGPT responds to the given prompt by posing contextual questions for further clarity. It also presents preliminary requirement recommendations in a table format, which might differ across rounds. Additionally, the table include requirements that ChatGPT autonomously suggests.

In the second following prompt clarifications and details to the additional questions are given as freeform. ChatGPT replies and generated refined output of requirements:

ChatGPT:” Thanks *for the information! Based on your responses, here is a revised Table of requirements:*

…

This Table outlines the system's main functional and non-functional requirements, allowing the developers to understand what they need to build. Additionally, it outlines sources of the requirements, which can be useful for reference and further consultation. “

EC2: ChatGPT offers a revised set of requirement specifications in a tabular layout. The specifications are determined by the details provided in the second prompt. The origin of these requirements is only specified in the initial round. Without the presentation of the source requirement, adjustments might be necessary to ensure traceability.

In the third prompt ChatGPT is asked to prioritize the requirements: “Can you prioritize requirements in the table?”. Now the output contains augmented table with additional column for prioritization (Must have, Should have, Could have, Won’t have).

EC3: In the table, requirements prioritization can be incorporated through an added prompt. This likely holds true for other custom adjustments as well. In the third iteration, ChatGPT introduced a prioritization for the requirements, even though it wasn't explicitly requested.

In the round 3 prompt iteration is demonstrated. Additional requests to the previous prompts can be made and ChatGPT responses to the changes.

EC4: ChatGPT reacts to prompt cycles according to user directions and is capable of grasping supplementary clarifications.

PEC1: ChatGPT is prompted to create requirements based on a given specification pattern. It is capable of seeking clarifications, offering recommendations, and producing high-level requirements presented in a tabular format in response to a prompt. However, the content of these requirements may differ depending on the round they were generated in.

5.1.2 Stakeholder and management support

Intention for this prompt pattern is to provide guidance and workflow how project manager could utilize ChatGPT when facing and communicating with stakeholders, project sponsors and provide more transparency and understanding to different project related challenges. Motivation for this pattern (Table 4) is to mitigate pain point where the project stakeholders do not understand the project objectives and how those are implemented in agile way. The prompt pattern is labeled as steering group meant for simulating the guidance and instructions provided by the project steering group.

Table 4. Steering group pattern

Contextual statements

`As a [roleA] your task is to fully understand agile principles, business objectives and give feedback about the project.`
`The project has the following objectives: [o1, o2, o3…].`
`Ask the necessary questions to fully understand the objectives and to be able to understand the project status and way of working.`
`Explain your understanding and provide necessary actions for me representing the [roleB].`
`Provide output in [format].`

Prompt pattern demonstration: “As a project’s steering group member your task is to fully understand agile principles, business objectives and give feedback about the project. The project has the following objectives: We are implanting an ERP integration system to customer using SAP product. Integration is done in agile way, and we expect to release one integration in every two weeks. At the moment we do not get customer to participate demos and would need their feedback. Project cannot start final testing before customer testing. Customer does not have time for the testing but is expecting the release soon. Ask the necessary questions to fully understand the objectives to be able to understand the project status and way of working. Explain your understanding and provide necessary actions for me as the project manager. Provide output in memo format.". Memo output is requested in the prompt as an example to give possibility to share the outcome using e.g., email. Its format has not been defined.

The first-round response of ChatGPT contains:*" …To further understand the situation and propose effective solutions, I need additional information on the following:*

1. *Customer Engagement: What are the current methods of customer engagement and how frequently are we reaching out to them for demos and feedback?*
2. *Customer Representative: Is there a dedicated customer representative involved in the project who can spare time for regular feedback and testing?*
3. *Testing: What is the nature and extent of testing required from the customer before we can move to final testing?*
4. *Release Expectations: What are the specific timeframes when the customer is expecting the releases?*
5. *Communication and Collaboration Tools: What tools are we currently using to facilitate communication and collaboration between our team and the customer..."*

EC5: During every prompt testing phase, ChatGPT demonstrates awareness of stakeholder and managerial backing. It displays comprehension of the challenges and seeks further information about client interactions, testing procedures, release anticipations, and methods of communication and collaboration. The provided information is formatted as a memo. However, in the second round of prompt testing, no additional questions were made by ChatGPT.

In the second prompt simulated additional information about requested questions is given to ChatGPT.

EC6: Upon clarification, ChatGPT offers recommendations on enhancing customer relationships in the given scenario. These suggestions serve as valuable guidance for addressing the issue. Nevertheless, the quantity and nature of the suggestions fluctuate based on the test pattern iteration. Typically, the response encompasses actions pertinent to the goals.

In test round 3 prompt iteration is demonstrated. Additional requests to the previous prompts can be made and ChatGPT responses to the changes.

EC7: ChatGPT reacts to iterative prompts according to user directives and has the capability to comprehend further explanations.

PEC2: ChatGPT recognizes its function as a member of the steering group and delivers presentation as requested. It appears to comprehend the topic at hand and offers actionable suggestions. These suggestions typically seem applicable. However, there is significant variability in the actions proposed during each round.

5.1.3 Role clarification

Clear role definition helps projects to overcome this pain point when there is uncertainty of the responsibilities and too much overlapping. Intention for this pattern is to provide guidance and workflow how project manager could utilize ChatGPT when defining the right roles for the project that help individuals to be accountable of specified tasks defined in the project.

To further elaborate, the avoidance of role voidance and overlap is crucial as it ensures that all necessary project tasks are accounted for and that team members are not burdened with multiple, conflicting responsibilities. This pattern (Table 5) helps to streamline the project process by assigning specific roles to individuals based on their skills and expertise. By doing so, team members can focus on their assigned tasks and contribute to the project's overall success.

Table 5. Role clarification pattern

Contextual statements

`As a [roleA] your task is to specify clear roles for the team.`
`The project team has the following constraints: [c1, c2, c3…].`
`Ask all necessary questions to fully understand the project and be able to create necessary role specifications.`
`Explain your understanding and provide solution to me representing the [roleB].`
`Provide output in [format].`

Prompt pattern demonstration: “As an engineering manager your task is to specify clear roles for the team. The project has the following constraints: software is created using C++ and publicly known open-source components. Only one person knows how to program in C++ but the rest 4 persons have experience only in Python and Javascript. Team needs to deliver C++ application in Linux environment in the next four months. Ask all necessary questions to fully understand the project and be able guide team to create necessary role specifications. Explain your understanding and provide solution to me representing the management. Provide output in Table format.”

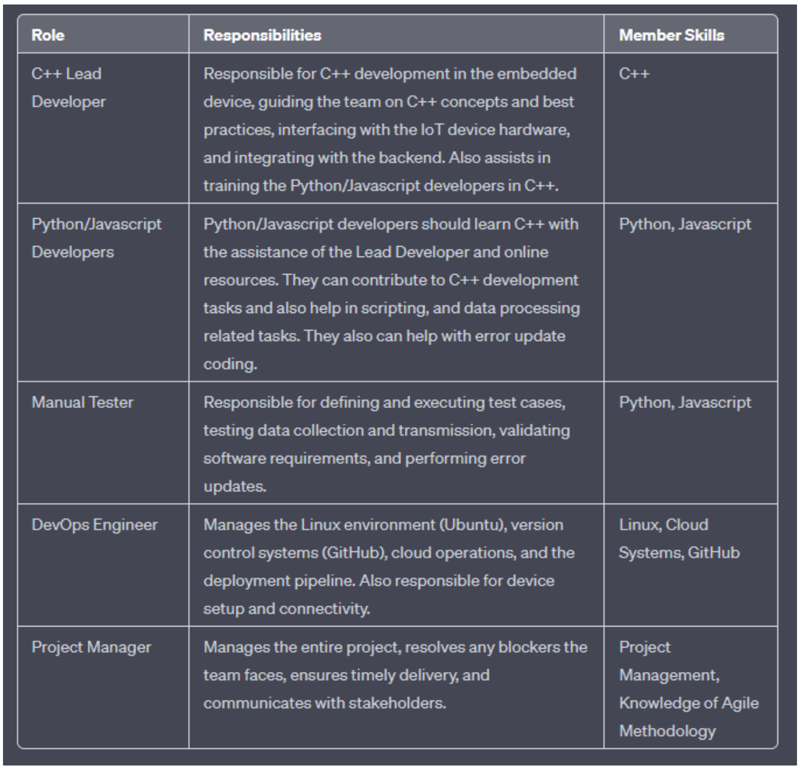
In the first round ChatGPT responses:” *Sure, I'd first need to understand a few more details about the project to make the role specifications as accurate as possible. Here are the questions…”*

EC8: ChatGPT appears to grasp the subject and context and offers supplementary questions for further clarity. In every response cycle, it lists initial roles along with their respective descriptions.

For ChatGPT’s response the following second prompt is provided to clarify the project role needs:” Project needs to deliver C++ application in embedded device and transfer the data to backend. It should collect IoT data and move that to the backend for further processing. We use existing cloud-based backend but IoT device as Atmel based 32-bit processor and necessary hardware. We would like to utilize existing sw designers also in C++ development. We collaborate through GitHub using its features. Linux is ubuntu based. Testing is done fully manually as we don't have suitable tools for testing C++ applications and the application is simple. We plan to make some error updates but otherwise maintenance is approx. two times in the year.”

EC9. ChatGPT processes the supplementary prompt, seeking clarifications on its queries. Once it assimilates the provided information, it then curates a detailed presentation outlining the necessary roles for the project. An illustration of ChatGPT's output can be viewed in Figure 16, reflecting its capability to synthesize and present complex project-related information.

ChatGPT: “*Based on the additional details, I suggest the following roles and responsibilities for your team members…”*



Role	Responsibilities	Member Skills
C++ Lead Developer	Responsible for C++ development in the embedded device, guiding the team on C++ concepts and best practices, interfacing with the IoT device hardware, and integrating with the backend. Also assists in training the Python/Javascript developers in C++.	C++
Python/Javascript Developers	Python/Javascript developers should learn C++ with the assistance of the Lead Developer and online resources. They can contribute to C++ development tasks and also help in scripting, and data processing related tasks. They also can help with error update coding.	Python, Javascript
Manual Tester	Responsible for defining and executing test cases, testing data collection and transmission, validating software requirements, and performing error updates.	Python, Javascript
DevOps Engineer	Manages the Linux environment (Ubuntu), version control systems (GitHub), cloud operations, and the deployment pipeline. Also responsible for device setup and connectivity.	Linux, Cloud Systems, GitHub
Project Manager	Manages the entire project, resolves any blockers the team faces, ensures timely delivery, and communicates with stakeholders.	Project Management, Knowledge of Agile Methodology

Figure 16. Roles specified by ChatGPT

ChatGPT*: “…The specifics of these roles might need to be adjusted based on the specifics of your team and your project, but this should give you a good starting point.”*

EC10: Based on the specific prompt test iteration, various role specifications are displayed. Moreover, in every cycle, ChatGPT underscores the potential need for modifications to the roles e.g: “*The specifics of these roles might need to be adjusted based on the specifics of your team and your project, but this should give you a good starting point.*”

In test round 3 prompt iteration is demonstrated. Additional requests to the previous prompts can be made and ChatGPT responses to the changes.

EC11: ChatGPT responses to prompt iterations based on user instructions and can understand additional clarifications.

PEC3: ChatGPT appears to grasp the context of the prompt pattern and presents an initial role description, which includes the role's responsibilities and necessary skills based on the provided feedback. Moreover, it conveys that the roles may require adjustments in accordance with the actual requirements of the project.

5.1.4 Redundancy analysis

Ensuring an overlap in skill sets necessitates proficient talent management and an ongoing commitment to learning so that varying skills can be cultivated as needed within a team. Intention of this prompt pattern is to give directions and aid project managers on how to utilize ChatGPT in spotting and addressing gaps in redundancy. Moreover, motivation for this pattern (Table 6) is formulating a competency development plan to combat the issues of insufficient redundancy. Teams might have specialists who excel in specific tasks but lack the cross-functional abilities necessary to provide a backup. This becomes problematic if the team needs to concentrate on tasks that only one member is proficient in.

Table 6. Redundancy analysis pattern

Contextual statements

`As a [roleA] your task is to` *analyze redundancy in agile team skills*`.`
`The project team has the following persons: [p1, p2, p3…].`
`Responsibilities of the team are: [r1, r2, r3…].`
`Ask the necessary questions to fully understand the task and create analysis of redundancy gaps in the team.`
`Explain your understanding and provide solution to me representing the [roleB].`
`Provide output in [format].`

Prompt pattern demonstration: “As a resource manager your task is to analyze redundancy in agile team skills. The project team has the following persons: one Java back-end programmer, test automation developer, two JavaScript front-end programmers and DevOps specialist. Responsibilities of the team are: create full-stack application for netbanking system, UI is based on JavaScript and backend is done using Java. Ask the necessary questions to fully understand the task and create analysis of redundancy gaps in the team. Explain your understanding and provide solution to me representing the project manager. Provide output in table."

ChatGPT response for the first-round pattern*: “Sure, to understand the skills and potential redundancy in this agile team, I would need to ask the following questions:*

What is the level of cross-skills of each team member?

Are any of the team members experienced or skilled in multiple roles?

What are the levels of proficiency in each team member's skill sets?

What specific responsibilities does each team member have in their respective roles?

How complex is the netbanking system the team is developing?

What is the estimated timeline for the project?

Are there any unique responsibilities or tasks that only specific team members can handle?

Do the team members have experience in working in agile teams and are they familiar with agile practices like pair programming, collective code ownership, etc.?

Do the team members have knowledge and skills in other relevant technologies that the project may require?

Based on these questions, we can analyze the redundancies and gaps in the team.…”

EC12: ChatGPT appears to grasp the intent behind the prompt and consistently poses supplementary questions during each testing phase of the prompt. This behavior is observed across all three pattern cycles, though there are slight variations in the quantity and substance of the questions.

In the second supplemental prompt additional information is given to ChatGPT. As there are several questions it is expected that project manager (roleB) does not necessarily have answers to each question. In the first test round response snippet ChatGPT is able to provide Table having roles, redundancy, potential gaps and solution proposal:

ChatGPT:” *Analysis*

Let's consider for our analysis that all the team members are proficient in their respective roles, they are experienced in agile practices, they have cross-skills but to a limited extent and there are no unique responsibilities that only specific team members can handle.

Given this, the redundancy analysis of the team would look something like this:

Role	*Redundancy*	*Potential Gaps*	*Solution*

Java Back-end Low High proficiency in Java Hire an additional Java developer or upskill the existing team members in Java

Test Automation Low High proficiency in Test Automation Upskill the existing team members in test automation

JavaScript Front-end Moderate High proficiency in Javascript No additional resources needed; however, upskill in JS could be beneficial

DevOps Low High proficiency in DevOps Upskill the existing team members in DevOps or consider hiring an additional DevOps specialist…"

EC13: Following the second prompt and upon receiving further details, ChatGPT conducts a gap analysis using the provided data. Even though the results differ considerably with each iteration of the same prompt, every outcome offers insightful recommendations worth considering.

In test round 3 prompt iteration is demonstrated. Additional requests to the previous prompts can be made and ChatGPT responses to the changes.

EC14: ChatGPT responses to prompt iterations based on user instructions and can understand additional clarifications.

PEC4: ChatGPT can discern the task within a prompt and offer various types of analyses and recommendations. When employing the same prompt pattern, the outcomes can significantly differ, presenting a wide array of thoughtful considerations.

5.1.5 Agile process understanding

Agile process understanding can be a pain point as participants do not necessarily have experience nor knowledge about it. This prompt pattern tries to help projects to overcome agile process understanding as a pain point.

Intention for this pattern (Table 7) is to provide guidance and workflow how project manager could utilize ChatGPT when struggling with the agile process implementation in the team. The actors in this pattern are the project manager and AI representing the agile coach. Motivation is to provide for the whole agile team coaching how agile process should be applied.

Table 7. Agile process coaching pattern

Contextual statements
As a [roleA] your task is to *coach agile processes*. The project has the following objectives: [o1, o2, o3…]. Ask the necessary questions to fully understand the objectives and create agile coaching plan to the team. Explain your understanding and show me the plan representing the [roleB]. Provide output in [format].

Prompt pattern demonstration. “As an agile coach your task is to coach agile processes. The project has the following objectives: build a web application to be ready in autumn market. We have backlog full of features. Some of the features are quite abstract. Job needs to be done in two weeks iterations and we should be able to test and release the software as often as possible. Project team is five persons with senior and junior designers not familiar of Scrum or agile ways of working. Ask the necessary questions to fully understand the objectives and create agile coaching plan to the team. Explain your understanding and show me the plan representing the project manager. Provide output in action plan format.”

In the first prompt test round ChatGPT responses:” Sure*, let's get started by asking a few clarifying questions:*

What roles do the five team members have? Are there dedicated developers, testers, or product owners?

Are there existing processes in place that the team follows, or are we starting from scratch?

What are the communication practices currently being used? Do they use any tools for communication or project management?

Can you clarify the scope of the application? Are there any technical specifications or constraints we should be aware of?

How familiar are the team members with the concept of Agile? Do they have any training or past experience with this methodology?

What is the timeline for the project? When is the expected launch date?

Who are the key stakeholders, and what are their expectations?

Understanding these aspects will help me create a more tailored Agile coaching plan.

Here's a draft of the Agile Coaching Plan considering the above details…"

EC15: ChatGPT seems to understand the prompt in each round and asks additional questions for clarification. There is variance in response in different test rounds of what kind of additional questions are asked.

For the second prompt additional information is given to ChatGPT. The answer is formulated based on the first-round response: "All team members are dedicated to their function. Team has tried to use Scrum but with no good luck. As being mostly remotely team has been used Teams with ad hoc meetings and some daily startup meetings. This is important project for the company as it is the first web application to the market. Timeline for project is the next four months. We would like to release after each sprint, but the sprint working is new. So, we don't have experience on Scrum practicalities. The only challenge in our team is the level of knowledge.".

ChatGPT responses for the second prompt with coaching plan which contains different kind of plans depending on the prompt test round.

EC16: Based on the feedback, ChatGPT can suggest an agile process coaching strategy. The nature of the plans presented varies across distinct prompt testing rounds.

In testing round 3 prompt iteration is demonstrated. Additional requests to the previous prompts can be made and ChatGPT responses to the changes.

EC17: ChatGPT responses to prompt iterations based on user instructions and can understand additional clarifications.

PEC5: ChatGPT appears to comprehend the context of the prompt as being based on agile coaching principles. Although the plan varies considerably, the proposed ideas remain beneficial. The coaching emphasizes ongoing learning and enhancement, with a well-outlined schedule presented.

PEC6: ChatGPT appears to grasp the prompt context, which is focused on adaptability. It's capable of formulating an action plan upon request and can use supplemental prompt details to fine-tune this plan. Even though the response to the same prompt can vary, it successfully identifies challenges and offers relevant recommendations.

5.1.6 Change resistance

Change resistance is identified as one pain point in agile projects [100]. When asked from ChatGPT it tells that people may resist change due to fear of the unknown, concerns about job security, or a preference for familiar processes provides [100]. Furthermore, eight strategies are proposed to overcome the change resistance: communicate the benefits, involve team members and stakeholders, provide training and support, address concerns and fears, implement change gradually, demonstrate success, foster a supportive culture, and celebrate milestones and achievements [100]. These kinds of strategies can be found also in literature. Kreutzer et al. talks about change management process having seven major barriers to a successful change management [102]:

- Misunderstanding of change necessity hinders implementation.
- Absence of change leader in top management reduces change acceptance.
- Inexperience in change processes complicates successful implementation.
- Insufficient skill set for new tasks slows transformation.
- Internal conflicts divert energy to unnecessary disputes.
- Lack of an adapted remuneration system signifies low management commitment.
- Resistance to change from management and staff forms a major obstacle.

The intention of the change resistance prompt pattern (Table 8) is to offer guidance to the agile team that is accustomed to traditional non-agile methods and finds the transition to agile workflows challenging. The motivation for this comes from the change management process, which can effectively guide newly formed agile teams.

Table 8. Change resistance pattern

Contextual statements
As a [roleA] your task is to *improve team to way of working to overcome change resistance.*
The project team has the following challenges: [c1, c2, c3…].
Ask the necessary questions to fully understand the challenges and provide plan how to overcome change resistance.
Explain your understanding and thinking to me representing the [roleB].
Provide output in [format].

Prompt pattern demonstration: “As an agile coach your task is to improve team way of working to overcome change resistance. The project team has the following challenges: team has been working in water-fall method over five years, scrum or agile way of working is new to the team, senior developers would like to continue the work as they have used to without dailies and retrospectives. Ask the necessary questions to fully understand the challenges and provide plan how to overcome change resistance. Explain your understanding and thinking to me representing the scrum master in the team. Provide output in action plan format.”

In the first prompt test round ChatGPT answer in the following way: *“Sure, in order to fully understand the challenges, I would need to ask some questions first:*

What specific issues or challenges is the team facing that have motivated this move towards Agile methodology? Is there any particular reason for the change?

How receptive is the rest of the team, aside from the senior developers, to this change? Are there any who are interested or have some experience with Agile?

Is there any support from the higher management for this transition? What resources, like training or coaching, are they willing to provide?

Have there been any previous attempts at Agile transformation, and if so, what were the outcomes and what lessons were learned?

How do the senior developers feel about change in general, and how do they handle it? What are their specific concerns about moving to Agile?

How does the team currently handle project management, and what tools do they use? Are there any elements of their existing system that could be used in a Scrum setting...”.

ChatGPT follows the pattern by requesting additional information for five questions.

EC18: ChatGPT seems to understand the context of the prompt in each round. In the first prompt test round it responds to the prompt by asking additional questions about teamwork. In latter two prompt test rounds these questions are not asked. Action plan is presented in response in each round.

In the second prompt more information is shared between ChatGPT and clarifications given using artificial project setup: "Scrum practicalities are difficult to follow. Team does not understand the importance of sprint planning and daily scrums. Team size is five persons plus product owner who is also having scrum master role. As the whole organization moved to use Scrum also this team needs to follow the process. Organization supports scrum masters but not individual team members. Senior developers won't buy the new thinking and want to stick in old way of working."

ChatGPT response in first round prompt:" Understood*. It seems like there's a larger organizational push for Scrum and Agile methodology, but the team, particularly the senior developers, are finding it hard to accept this change. The issues of not understanding the importance of sprint planning and daily scrums, coupled with the lack of organizational support for individual team members, are also significant challenges*…"

EC19: ChatGPT provides revised action plan in each round. There is much variance in actions plans generated by each round. However, action plans contain good coaching options how to lower the resistance of change in teamwork.

In round 3 prompt iteration is demonstrated. Additional requests to the previous prompts can be made and ChatGPT responses to the changes.

EC20: ChatGPT responses to prompt iterations based on user instructions and can understand additional clarifications.

PEC7: ChatGPT appears to comprehend the prompt pattern related to resistance to change and can devise an action plan grounded on user input. This plan encompasses strategies to tackle resistance to change, although the responses may vary when the same prompt is used multiple times.

5.1.7 Adaptability understanding

Intention for this prompt pattern (Table 9) is to coach team to respond effectively to different changes and adapt their working style to correspond to the project objectives. Motivation is to challenge team to reflect what works and what does not and rapidly change the way of working. In other word to continuously improve the way of working. Adaptability in this case is similar to team's agile retrospective. Use AI may help coaching team to adapt to the new ways of working and performing AI assisted retrospective and create the necessary second option actions for change.

Table 9. Adaptability management pattern

Contextual statements

`As a [roleA] your task is to` *coach team to adapt and motivate in occurring changes*`.`
`The project has the following objectives: [o1, o2, o3…].`
`Ask the necessary questions to fully understand the objectives and create a plan how to adapt to the changes.`
`Explain your understanding and provide the actions to me representing the [roleB].`
`Provide output in [format].`

Prompt demonstration: "As an agile coach your task is to coach team to adapt and motivate in occurring changes. The project has the following objectives: to deliver ERP system integration to new vendor system, vendor is changing their way of working constantly, team has used to work in two weeks sprints, but the customer is working ad hoc and does not respect sprints. Ask the necessary questions to fully understand the objectives and create a plan how to adapt to the changes. Explain your understanding and provide the actions to me representing the project manager. Provide output in Table."

In the first test round prompt demonstration ChatGPT responses:" *I understand the challenges that the team is facing - dealing with a vendor that changes its approach constantly, a customer working in an ad hoc fashion, and the disruption to your usual two-week sprint cadence. As an Agile coach, it's important to understand the context deeply to create an effective plan. Here are a few questions I would ask to get a better understanding…*"

EC21: ChatGPT seems to understand the prompt context in each round and is expecting clarifications for creating an action plan. Multiple questions are presented, and their count and content depend on the prompt test round. In addition, preliminary plan is presented.

In the second prompt additional information is given to the ChatGPT: “Vendor changes happens quite ad hoc but typically in week or two. ERP used is SAP and persons have some knowledge of that as end users. We have weekly meetings over Teams with the vendor. Customer is not aware of vendor schedule as there is not direct channel between them. Customer is expecting releases on time independently on agile system used. We stick in the own release cycle but there is pressure to change that towards customer's schedule. Team is not motivated when lots of unplanned changes are coming.”.

ChatGPT’s response after the second prompt in round one*: “Thank you for providing the additional information. I understand your concerns and see the challenges the team is facing with the frequent changes from the vendor, the customer's demand for on-time delivery, and the impact on the team's motivation. Based on your responses, I am refining the action plan as follows…”*

EC22. After second complementary prompt ChatGPT creates list of actions. These actions have variation depending on the prompt test round but contain suggestions for the mentioned items in the prompt.

In round 3 prompt iteration is demonstrated. Additional requests to the previous prompts can be made and ChatGPT responses to the changes.

EC23: ChatGPT responses to prompt iterations based on user instructions and can understand additional clarifications.

5.1.8 Maintaining agile way of working

ChatGPT is asked to give an opinion on maintaining agile way of working. Although this can be seen as a major pain point ChatGPT thinks is may not: “Maintaining the agile way of working and avoiding a return to old practices can be challenging, especially in organizations with deeply ingrained traditional processes. While this may not be a major pain point in every agile project, it is a critical aspect of sustaining agile adoption and ensuring the long-term success of agile initiatives.” [100]

Anyhow, motivation for maintaining agile way of working is seen important and this pattern (Table 10) is used to provide tools to not to revert team way of working back to the old style. Intention is to find way how ChatGPT can be utilized in this manner.

Table 10. Agile way of working pattern

Contextual statements
`As a [roleA] your task is to coach team to maintain agile way of working.`
`The project has the following challenges: [c1, c2, c3…].`
`Ask the necessary questions to fully understand the challenges and create coaching plan to maintain agile way of working.`
`Explain your understanding and provide plan to me representing the [roleB].`
`Provide output in [format].`

Prompt pattern demonstration: “As a scrum master your task is to maintain agile way of working. The team has the following challenges: Team does not do proper sprint planning, no retrospectives are made, daily scrums do not have value for the team, team is changing the process without any change management. Ask the necessary questions to fully understand the challenges and create coaching plan for maintaining agile way of working. Explain your understanding and provide plan to me representing the project manager. Provide output in notes.”

ChatGPT response in the first round: “*As a Scrum Master, it's essential that I fully understand the team's current situation, perspectives, and the challenges we're facing. To do so, I will need to ask several questions. Let me outline them first…”*

EC24: For the prompt pattern ChatGPT understands its role as Scrum Master and request additional information in each prompt test round.

In the second additional prompt the following artificial information is described:” Team does not see the point in sprint planning as senior developers think they can do it individually. Each person already has some role and no cross-team roles are not seen important. I as senior developer do not see any big difference working as we have used to.”

In first round ChatGPT responds to the second prompt: “*Thank you for sharing your perspective. As a Scrum Master, it's my responsibility to ensure that everyone understands the benefits of Agile processes, so let's discuss some of these concerns…*”. It also provides concerns about sprint planning and cross-functional roles.

EC25: ChatGPT acts for the second prompt by identifying the key elements which are here sprint planning and cross-functional roles challenges.

In test round 3 prompt iteration is demonstrated. Additional requests to the previous prompts can be made and ChatGPT responses to the changes.

EC26: ChatGPT responses to prompt iterations based on user instructions and can understand additional clarifications.

PEC8: ChatGPT understands the principles of agile processes and the roles associated with them, including the Scrum Master's responsibilities. Once it has grasped the context, it elaborates on the importance of sprint planning and cross-functional roles. The primary intent of the original prompt appears to target a senior developer, likely to enlighten and encourage them in their duties.

5.1.9 Effort estimation

In agile projects as in any project, effort estimation may constitute a significant challenge due to the inherent difficulties in precisely quantifying the required effort for tasks. This issue is exacerbated by the iterative characteristics of agile methodologies, coupled with the likelihood of evolving requirements. Consequently, imprecise estimations could result in misaligned expectations, suboptimal resource distribution, and project delays. [100]

Intention for this prompt pattern (Table 11) is to provide effort estimation for the project. Motivational part of effort estimation comes from the fact that effort estimation is probably one of the difficult tasks in the project management.

Table 11. Effort estimation pattern

Contextual statements
As a [roleA] your task is to *create an effort estimation*.
The user story has the following requirements: [r1, r2, r3…].
Ask the necessary questions to fully understand the requirements and create effort estimation and required tasks.
Explain your understanding and provide effort estimation to me representing the [roleB].
Provide output in [format options].

Prompt pattern demonstration:” As a senior developer your task is to create an effort estimation. The user story has the following requirements: User needs to login to the system, she needs to enter her email address and password, password needs to be unique and is stored in to nosql db, password needs to be changed every 30 days and cannot be reused. Ask the necessary questions to fully understand the requirements and create effort estimation and required tasks. Explain your understanding and provide effort estimation to me representing the scrum master. Provide output in Table containing tasks and effort needed.”

In the first prompt test round ChatGPT responses:” Sure*, before giving an estimation, I'd like to clarify a few points:*

1. Authentication system: Do we already have an existing authentication system that we can leverage, or do we need to create one from scratch?

2. User interface: Will I be developing the UI for the login, registration and password change functionality, or will it be provided?

3. NoSQL DB: Which specific NoSQL database are we using? MongoDB, DynamoDB, Couchbase?

4. Uniqueness of the password: When we say the password needs to be unique, does it need to be unique for each user or unique globally, i.e., no two users can have the same password?

5. Password change reminder: How should the user be reminded to change their password every 30 days? Via email or any other medium?

6. Password reuse policy: How many previous passwords should the system remember to prevent reuse...”

EC27: For the prompt pattern ChatGPT seems to understand the requirement and provides additional questions for clarification in each prompt test round.

In the second prompt more artificial project information is given to ChatGPT:” We use existing AA in backend which can be configured for our purpose. So, the topic is actually only designing the web UI. Password has minimum length of 12 characters and should have at least two special characters and two capital letters. Uniqueness is stored in the backend so nobody can use password used once. Email verification is used. All passwords since the beginning are stored. NoSql will be Redis as it is fast and scalable. We use three Redis nosql nodes. Passwords are hashed using HMAC and salting is also needed.”.

ChatGPT provides response to the second prompt in the first round: “*Thank you for the additional information. Given these details, it sounds like most of the backend functionality is already in place. We are primarily tasked with designing the front-end* *interface and integrating it with the backend. Additionally, we will need to handle the password validation requirements on the front end.*

Here's an updated task breakdown and effort estimation, taking into account these new details…"

EC28. In response to the second prompt, ChatGPT provides tasks and estimates for each system component. While it offers valuable recommendations, the projected work volume may be inflated, as there's a noticeable discrepancy across different prompt test iterations.

In round 3 prompt iteration is demonstrated. Additional requests to the previous prompts can be made and ChatGPT responses to the changes.

EC29: ChatGPT responses to prompt iterations based on user instructions and can understand additional clarifications.

PEC9: ChatGPT appears to comprehend the motivation behind an effort estimation prompt. It supplies a task list it deems necessary, complete with descriptions and time estimates in days. It also explains the rationale behind the given estimates and their potential impact on the effort required. However, the estimation significantly varies when the question is repeated.

5.1.10 Technical knowledge

The project team might not possess the required technical expertise to carry out project tasks. This will have an effect in accuracy of effort estimation. The intention of this prompt pattern (Table 12) is to evaluate the availability of essential skills and match the right development skills in team. As agile teams are self-organizing this might not always be the best way to allocate tasks. The motivation for this prompt structure is to assist project management and team to match planned tasks against right skillset and use AI assistance to provide SWOT analysis.

Table 12. Technical knowledge analysis pattern

Contextual statements

`As a [roleA] your task is to estimate technical knowledge in team to support effort estimation.`
`The project has the following` *technical objectives*`: [o1, o2, o3…].`

Ask the necessary questions to fully understand the *technical objectives* and create plan how team can fulfill the needed technical skills.

Explain your understanding and provide plan for me representing the [roleB].

Provide output in [format].

Prompt pattern demonstration: “As an engineering manager your task is to estimate technical knowledge in team to support effort estimation. The project has the following technical objectives: Backend is developed using Java and SQL database, system is running on top of RedHat Linux, we need some sort of DevOps, we should need about 3-4 man months to do the project, existing code base exists so only new functionality need to be developed. Ask the necessary questions to fully understand the technical objectives and create plan how team can fulfil the needed technical skills. Explain your understanding and provide plan for me representing the project manager. Provide output in memo format.”

In the first prompt test round ChatGPT responses the following:

“*Subject: Technical Knowledge Estimation for Upcoming Project*

Dear [Project Manager],

I hope this message finds you well. In response to your request, I have reviewed the project objectives and have identified key areas where we need to assess our team's technical capabilities. Here are the necessary inquiries to fully understand the technical objectives:

1. *Java: What is the specific experience level with Java within the team? Are there any specialized libraries or frameworks (Spring Boot, Hibernate, etc.) used in our existing codebase that the team should be familiar with?*
2. *SQL Database: What is the team's proficiency in SQL? Are there any specific SQL platforms we're using (MySQL, PostgreSQL, Oracle, etc.)? Are there any existing database schemas or specific database management tasks the team should be familiar with?*
3. *RedHat Linux: How comfortable is the team with Linux system administration? Specifically, what is their experience with RedHat Linux and its server management?*

4. DevOps: What DevOps practices and tools (Continuous Integration/Continuous Delivery, Infrastructure as Code, Docker, Kubernetes, Jenkins, etc.) are currently in place? What is the team's proficiency in these areas?

5. Project Scale: Is the 3-4 man-month estimate based on previous projects of similar scale? Does this estimate consider potential challenges or risks that might arise?

6. Existing Codebase: How familiar is the team with the existing codebase? Will there be any refactoring needed to implement the new features?"

EC30: ChatGPT identifies the technical objectives presented and asks additional questions for clarifications in each prompt test round.

In the second prompt additional information is given to ChatGPT using artificial project information: "The integration is using CSV, XML and ASCII files between the systems. These are transferred using sftp. Team has two software engineers and they know the basics but not have done this before. Project should be ready in three months. We don't have any training resources and team members need to browse internet and ask from Google."

EC31: ChatGPT fills the memo based on answers given in the second round.

ChatGPT response after first prompt test round: "…*Here is an updated plan to ensure our team acquires and enhances the necessary technical skills:*

1. Self-learning and Research: Encourage our team members to leverage online resources for learning. There are various reputable websites, blogs, forums, and video tutorials available for free, where they can gain deeper insights into file integration using CSV, XML, and ASCII, and transferring files using sftp.

2. Structured Learning Path: Guide the team members to follow a structured learning path. A suggested path would be to start with a deep understanding of working with CSV, XML, and ASCII files, followed by sftp. Then move on to integration techniques involving these file types.

3. Community Participation: Encourage participation in developer communities like StackOverflow, GitHub, etc., where they can pose their doubts and interact with experienced developers. This way, they will not only find solutions to their issues but also get to learn from experts.

4. Practical Implementation: Encourage them to work on mini projects or tasks that involve the use of these technologies. This will give them a hands-on experience and help them understand the practical challenges better.

5. Peer Learning: Promote a collaborative learning environment. The team members can share their learnings and solutions to the problems they faced during their self-learning journey. This can be done through weekly knowledge-sharing meetings.

6. Project Timeline: Given the learning curve and the 3-month project timeline, I recommend starting with simpler tasks related to the project, gradually moving to complex tasks as our team members get more comfortable with the technologies. This would require a detailed project plan taking into consideration the learning and upskilling required."

EC32: ChatGPT offers a revised plan featuring various recommendations to boost the technical abilities of the team. The content and advice within the plan differ significantly based on the specific testing round.

In test round 3 prompt iteration is demonstrated. Additional requests to the previous prompts can be made and ChatGPT responses to the changes.

EC33: ChatGPT responses to prompt iterations based on user instructions and can understand additional clarifications.

PEC10: ChatGPT recognizes the prompt as one addressing the management of technical knowledge within a team. It generates a proposal for how the team might surmount this challenge, tailored to the additional information provided in the prompt. Interestingly, different rounds of the same prompt yield varying suggestions.

6. DISCUSSION

This chapter delves into the academic and applied effects analyzed in this thesis. The academic influence is sourced from the outcomes of empirical research, and their pertinence is weighed against the literature review on agile methodologies and their associated difficulties. In the following part, the dialogue on the applied impact furnishes recommendations for addressing empirical discoveries utilizing patterns in ChatGPT prompts.

6.1 Theoretical implications

The primary empirical contribution (PEC) derived from the study are viewed as the fundamental findings. These are compiled and displayed in Table 13. Each PEC is a consolidation of two or more empirical contributions (EC), distilling the observation into a single entity. The initial contribution (PEC1) employs a requirement specification prompt pattern with ChatGPT. The resulting requirement specification adheres to a structure that includes a unique ID, description, and priority, all arranged in a Table as requested. In agile methodologies, requirements are usually assembled as user stories [103]. Nonetheless, this type of format could be attained by directing ChatGPT to produce requirements in the form of user stories. Agile principles can be incorporated into requirement management [82]. By fine-tuning the prompt requirements, they can be structured in a product backlog style.

In the second contribution (PEC2), ChatGPT recognizes the stakeholder and management pattern and identifies itself as a member of the steering group, addressing the challenges provided. This pattern explores the hurdles involved in acquiring and sustaining stakeholder and management support in an agile setting, along with suggested tactics to mitigate these issues. The response to the prompt presents a variety of solutions for the specified example challenges and is provided in the requested memo style. The suggestions offered by ChatGPT are in accordance with agile project practices documented in the literature [3].

The definition of roles is essential in an agile project [85]. PEC3 presents ways in which the project can receive assistance in clarifying roles and suggestions. At a theoretical level, ChatGPT offers similarities and diverse instructions for agile projects. As an agile team might have specific roles, it's vital to ensure a certain level of redundancy to accommodate scenarios where team members might leave, or other situations arise. PEC4 demonstrates how ChatGPT can contribute ideas to maintain redundancy within an agile project team. Redundancy is an important aspect in self-organizing teams and fostering flexibility [84]. In this regard, ChatGPT can assist teams and projects by providing agile guidelines.

Project teams may have difficulties in understanding agile principles or guidelines. PEC5 shows how ChatGPT can be used to give guidelines to the project using ways familiar in the literature. Human factor has been identified as the weakest link in the organization chain [89]. It is important that agile process understanding is priority for project team members and for the whole organization.

Table 13. Contributions in relation to literature

Id	Empirical contribution	Literature reference
PEC1	ChatGPT is programmed to create requirements based on a given specification pattern. It is capable of seeking clarifications, offering recommendations, and producing high-level requirements presented in a tabular format in response to a prompt. However, the content of these requirements may differ depending on the round they were generated in.	Supports, created requirements can be created to help clarify their content and traceability [3] [82].
PEC2	ChatGPT recognizes its function as a member of the steering group and delivers presentation as requested. It appears to comprehend the topic at hand and offers actionable suggestions. These suggestions typically seem applicable. However, there is significant variability in the actions proposed during each round.	Supports, information generated can be used to deliver data supporting stakeholder and management engagement [104].
PEC3	ChatGPT appears to grasp the context of the prompt pattern and presents an initial role description, which includes the role's responsibilities and necessary skills based on the provided feedback. Moreover, it conveys that the roles may require adjustments in accordance with the actual requirements of the project.	Supports, defining and clarifying the roles is essential in agile projects [85].
PEC4	ChatGPT can discern the task within a prompt and offer various types of analyses and recommendations. When employing the same prompt pattern, the outcomes can significantly	Supports, analyzing redundancy in team is important to guide members to support each other’s work [84].

	differ, presenting a wide array of thoughtful considerations.	
PEC5	ChatGPT appears to comprehend the context of the prompt as being based on agile coaching principles. Although the plan varies considerably, the proposed ideas remain beneficial. The coaching emphasizes ongoing learning and enhancement, with a well-outlined schedule presented.	Supports, coaching can improve agile process understanding [3].
PEC6	ChatGPT appears to grasp the prompt context, which is focused on adaptability. It's capable of formulating an action plan upon request and can use supplemental prompt details to fine-tune this plan. Even though the response to the same prompt can vary, it successfully identifies challenges and offers relevant recommendations.	Supports, proposals to organizations to perform at a high level possess the ability to adapt, adjust, and innovate on an ongoing basis [3] [88].
PEC7	ChatGPT appears to comprehend the prompt pattern related to resistance to change and can devise an action plan grounded on user input. This plan encompasses strategies to tackle resistance to change, although the responses may vary when the same prompt is used multiple times.	Supports, coaching teams to understand continual change and dedication necessity [72] [102][105].
PEC8	ChatGPT understands the principles of agile processes and the roles associated with them, including the Scrum Master's responsibilities. Once it has grasped the context, it elaborates on the importance of sprint planning and cross-functional roles. The primary intent of the original prompt appears to target a senior developer, likely to enlighten and encourage them in their duties.	Supports, coaching to make change to permanent without reverting back to the old way of working [3] [100].
PEC9	ChatGPT appears to comprehend the motivation behind an effort estimation prompt. It supplies a task list it deems necessary, complete with descriptions and time estimates in days. It also explains the rationale behind the given estimates and their potential impact on the effort required. However, the estimation significantly varies when the question is repeated.	Supports, suggesting effort estimations can help in misaligned tasks and project delays [3] [100].
PEC10	ChatGPT recognizes the prompt as one addressing the management of technical knowledge within a team. It generates a proposal for how the team might surmount this challenge, tailored to the additional information provided in the prompt. Interestingly, different rounds of the same prompt yield varying suggestions.	Supports, suggestions how to gain skilled resources as they play a crucial role in the project's outcome [3] [89].

In the sixth primary empirical contribution (PEC6), ChatGPT recognizes prompt pattern for adaptability and provides coaching instruction for the project. Organizations performing at a high level possess the ability to adapt, adjust, and innovate on an ongoing basis [88]. These instructions follow the agile way of working practices found in the literature as in agile practice guide [3].

In the seventh primary empirical contribution (PEC7), change resistance prompt pattern is demonstrated. ChatGPT identifies the change resistance pattern and provides different coaching solutions. The challenges associated with addressing change resistance within teams and present strategies for overcoming these issues are provided. To become a lasting and integrated part of teams and the organization, continual change and dedication are necessary [72]. The recommended strategies are based on agile principles and aim to create an environment that embraces change [3].

In the eighth primary empirical contribution (PEC8), ChatGPT recognizes the prompt pattern for maintaining an agile way of working, particularly in organizations with a history of traditional processes. As response provided strategies align with typical agile processes, such as reinforcing agile values and principles, continuous learning, and having regular retrospectives. Once again these follow agile practices such in agile practice guide [3].

In the nineth primary empirical contribution (PEC9), effort estimation prompt pattern is demonstrated. ChatGPT identifies effort estimation as a potential pain point in agile projects, primarily due to changing requirements and the iterative nature of agile methodologies. Impact of bad effort estimation is understood and strategies for addressing these issues are presented. Recommendations include using relative estimation techniques, breaking tasks into smaller units, and leveraging historical data. Imprecise estimations could result in misaligned expectations, suboptimal resource distribution, and project delays [100].

In the last primary empirical contribution (PEC10), ChatGPT recognized technical knowledge as a potential pain point in agile projects. The importance of technical knowledge in agile projects cannot be overstated. The challenges associated with addressing technical knowledge gaps in agile teams is identified and strategies for overcoming these issues is presented. Recommendations include encouraging cross-functional collaboration, hiring experts, and fostering continuous learning. Although technical knowledge has not been tied directly into work estimation it helps planning and implementation in the project. Skilled resources as they play a crucial role in the project's outcome [89].

6.2 Practical implications

Table 14 contains proposals to primary empirical contribution that are based on the responses of ChatGPT and pain point analysis. These proposals offered are made based on the evaluation of agile project practices and theoretical background.

The practical consequences of ChatGPT's initial primary empirical contribution (PEC1) reveal that requirements are produced at a high level. Nevertheless, these requirements vary depending on the prompt iteration, meaning that using the same pattern multiple times yields different types of requirements. In practical terms, the output can serve as a starting point for requirements. Furthermore, the prompt pattern should be more explicitly defined to guide how the requirements should be represented and iteration should be used to fine-tune the requirements.

Effort estimation (PEC9) and technical knowledge (PEC10) are related to the work estimation seen as one pain point. When applying prompt pattern for both are the responses quite generic ones.

Table 14. Proposals for using ChatGPT

Id	Empirical contribution	Recommendation
PEC1	ChatGPT is programmed to create requirements based on a given specification pattern. It is capable of seeking clarifications, offering recommendations, and producing high-level requirements presented in a tabular format in response to a prompt. However, the content of these requirements may differ depending on the round they were generated in.	Using pattern related, ChatGPT can produce the preliminary requirements, and the results might also include supplemental requirements. The iterative prompt method can be utilized to refine the list of requirements.
PEC2	ChatGPT recognizes its function as a member of the steering group and delivers presentation as requested. It appears to comprehend the topic at hand and offers actionable suggestions. These suggestions typically seem applicable. However, there is significant variability in the actions proposed during each round.	Pattern related allows ChatGPT to provide guidance on coaching for steering groups. To achieve more consistent suggestions, it's recommended to use more detailed prompts. Engaging in iterative prompting can help enhance the quality and

		precision of the initial prompt's content.
PEC3	ChatGPT appears to grasp the context of the prompt pattern and presents an initial role description, which includes the role's responsibilities and necessary skills based on the provided feedback. Moreover, it conveys that the roles may require adjustments in accordance with the actual requirements of the project.	With pattern related, ChatGPT can propose preliminary roles, though these should be considered as illustrative and may not directly translate to actual project settings. Using iterative prompting can aid in honing the substance of the starting prompt.
PEC4	ChatGPT can discern the task within a prompt and offer various types of analyses and recommendations. When employing the same prompt pattern, the outcomes can significantly differ, presenting a wide array of thoughtful considerations.	With pattern related, ChatGPT is capable of generating ideas to tackle redundancy challenges. It would be beneficial to review and analyze the results meticulously, perhaps with several iterations. Utilizing prompt iteration can assist in enhancing the clarity of the initial prompt.
PEC5	ChatGPT appears to comprehend the context of the prompt as being based on agile coaching principles. Although the plan varies considerably, the proposed ideas remain beneficial. The coaching emphasizes ongoing learning and enhancement, with a well-outlined schedule presented.	With pattern related, ChatGPT offers various options for agile coaching. The technique of prompt iteration can be employed to refine the content of the initial prompt.
PEC6	ChatGPT appears to grasp the prompt context, which is focused on adaptability. It's capable of formulating an action plan upon request and can use supplemental prompt details to fine-tune this plan. Even though the response to the same prompt can vary, it successfully identifies challenges and offers relevant recommendations.	With pattern related, ChatGPT suggests methods to help a team adapt to changes, serving as valuable tips for an agile way of working. Iterating the prompt to refine the results could be beneficial. Using prompt iteration can assist in enhancing the clarity of the initial prompt.
PEC7	ChatGPT appears to comprehend the prompt pattern related to resistance to change and can devise an action plan grounded on user input. This plan encompasses strategies to tackle resistance to change, although the	With pattern related, ChatGPT offers valuable suggestions for measures to mitigate resistance to change in agile projects. Iterating the prompt to

	responses may vary when the same prompt is used multiple times.	refine the action plan could be beneficial. The technique of prompt iteration can be used to improve the clarity of the initial prompt.
PEC8	ChatGPT understands the principles of agile processes and the roles associated with them, including the Scrum Master's responsibilities. Once it has grasped the context, it elaborates on the importance of sprint planning and cross-functional roles. The primary intent of the original prompt appears to target a senior developer, likely to enlighten and encourage them in their duties.	The suggested prompt pattern can assist with agile coaching. It's recommended to finetune the prompt pattern for specific purposes. Using the process of prompt iteration can aid in refining the content of the initial prompt.
PEC9	ChatGPT appears to comprehend the motivation behind an effort estimation prompt. It supplies a task list it deems necessary, complete with descriptions and time estimates in days. It also explains the rationale behind the given estimates and their potential impact on the effort required. However, the estimation significantly varies when the question is repeated.	The proposed prompt pattern can be utilized to assist with estimation tasks. It's advisable to finetune the prompt pattern for unique applications, keeping in mind that the risk of giving accurate estimation is high. The technique of prompt iteration can be employed to enhance the clarity of the initial prompt.
PEC10	ChatGPT recognizes the prompt as one addressing the management of technical knowledge within a team. It generates a proposal for how the team might surmount this challenge, tailored to the additional information provided in the prompt. Interestingly, different rounds of the same prompt yield varying suggestions.	The provided prompt pattern suggestion can be instrumental in analyzing the technical skills of a team. It's recommended to finetune this prompt pattern for specific uses, though the risk of hallucination is considerable. Utilizing prompt iteration can be beneficial in refining the content of the initial prompt.

Suggestions for the prompt patterns are given in separate cheat sheet which can be found in the appendix D. ChatGPT (GPT-4) has limitation in prompts having token limitation of 2048 [100]. This may cause inaccuracy in prompt handling.

7. CAPABILITIES OF CHATGPT

In the early months of 2023, the technological landscape witnessed rapid developments in generative AI technologies. Foremost among these innovations was ChatGPT and its suite of applications, epitomizing the pace at which these technologies have evolved. However, such a vibrant and fast-evolving sector presents challenges in comprehensive documentation. As a result, this study can, at best, offer a snapshot of the prevailing trends and advancements at this juncture.

Several Agile frameworks exists for large- and small-scale projects. Besides, each agile project can be individual creating specific challenges. Literature research in this study has shown some of the challenges can be recurring and common for different projects. These can be called typical pain points in agile projects. Although the literature research highlights pain points in agile project such as requirements management, stakeholder and management support, role definition, redundancy, agile process understanding, adaptability, change resistance, sustaining agile practices, work estimation, and technical knowledge, it's evident that there might be other hurdles tailored to certain teams, sectors, or organizational environments.

Agile methodologies mentioned in this study, celebrated for their adaptability, and focus on ongoing evolution, have kept pace with the broader technological advancements. This study delved into the fusion of AI technologies with agile project management methods and related pain points. Interestingly, the marriage of the two is still in its infancy. Presently, the intersection is mostly observed in the conversational application of AI. A testament to the prowess of these systems can be seen in GenAI LLMs like ChatGPT, which can craft and supply diverse valuable data through the art of prompt engineering.

A key finding from all primary empirical contributions is that ChatGPT can offer extensive guidance and instructions that are grounded in the reference literature. This implies a practical application wherein ChatGPT can supply templates and strategies for managing certain project pain points. By using explicit prompt engineering and design patterns ChatGPT responses can be narrowed down the information to be relevant to the given problem and help in managing or supporting generic agile project pain points. Nevertheless, differences were evident in the reactions when identical prompt templates were used in various cycles. In real-world terms, this implies having multiple response alternatives which the reviewer would need to accept. Additionally, tweaking each prompt or undergoing further iterations may offer deeper insights into the requirements of each situation. The phenomenon referred to as "hallucination" wasn't encountered with the prompts used, and the produced information aligned with literature research. It was found that NLG models frequently produce text that either doesn't make sense or deviates from the given source input. Such unintended outputs from the models have been termed by researchers as "hallucination.".

ChatGPT proves to be an invaluable asset for contemporary project management, offering valuable support in addressing and navigating challenges within an agile project. While it offers established theoretical expertise in agile project management, its role remains primarily at a high-level, leaving still communication, problem-solving, and strategic decision-making to the project team itself. With its ability to provide guidance and insights, ChatGPT augments project management practices by equipping teams with a reliable source of theoretical knowledge, enabling them to tackle hurdles effectively. However, the ultimate responsibility for executing effective communication, problem-solving, and strategic planning still lies with the project team.

7.1 Answers to research questions

In this chapter explicit answers are given to the research questions (RQs).

RQ1: What are the typical pain points of agile project?

Based on the literary review detailed in section 3.1, common pain points in agile methodologies include issues with requirement management, stakeholder and managerial backing, clarity in roles, team member’s overlap, understanding of agile processes, adaptability to various shifts, reluctance to change, sustaining agile practices, effort prediction, and adequate technical expertise. Each of these challenges can impact agile projects and their oversight. Section 3.2's Table 1 categorizes these agile project challenges and cites literature sources that validate these observations. These challenges can be grouped under categories such as project, individuals, processes, persistence, and effort estimation.

RQ2: How GenAI could guide to overcome typical pain points of agile projects?

GenAI (ChatGPT) can provide valuable insights and guidelines that are in line with those found in literature research. Specific prompt patterns can shape the responses of ChatGPT, facilitating the creation of reviews and recommendations for handling project challenges. ChatGPT carries significant potential as an auxiliary tool in agile projects, addressing common pain points. It can draw from its historic data to benchmark and recognize similarities in each project, thereby assisting in anticipating issues already experienced in its pre-trained data. However, while ChatGPT can indeed contribute helpful suggestions for managing pain points, it remains crucial that the project team maintains proficiency in agile methodologies and doesn't wholly depend on the GenAI's responses without critical assessment. Chapter 5 illustrates how the Primary Empirical Contributions (PEC) affirm this comprehension, showcasing the effectiveness of prompt patterns in supplying intelligence solutions to address pain points in agile project management.

RQ3: Can GenAI used in project management assistance such as planning, reporting, and estimating?

This study, which incorporates discussions with ChatGPT using prompt patterns, illustrates its utility in generating a range of templates and preliminary plans for agile projects. ChatGPT, being a pre-trained language model, has a foundational understanding of various agile project types thanks to the information it was trained on. However, given the inherent variations in project requirements and human resources, a blanket trust in the information generated by ChatGPT is not advisable. It's crucial to exercise caution and subject ChatGPT's outcomes to review by agile project specialists, as they may not be universally reliable. The outcome of PECs (chapter 5) and prompt pattern demonstrations according to Appendices A, B, and C give examples that ChatGPT can be helpful hand in various project management related tasks. As stated in Appendix C, demonstrations of prompts with additional iterations can be employed to enhance and direct the responses of ChatGPT.

7.2 Limitations of research

The study is generally limited to utilizing GenAI in addressing the pain points of agile projects. The pain point model has been chosen based on a literature review, but there may be other such pain points. The prompts used in the study aim to simulate various project management situations by providing fictional examples related to the pain point model. In this regard, they are based on the author's personal experience and simulation, with the intent to challenge ChatGPT with different prompts. ChatGPT accepts long sentences and various types of inputs, and the study has aimed to create as clear and simple prompts as possible so that ChatGPT's response can be reflected upon in relation to different pain points.

During the DSR for prompt engineering, the GPT-4 model had a constraint of 25 messages every 3 hours, and with the release of different versions, there might have been some discontinuities in chat interactions. Moreover, while actual projects weren't directly used to validate prompt patterns, example prompts did incorporate some anonymized project-related challenges.

Conversations i.e., context window might have limited different answers. ChatGPT has had a ubiquity definition for conversational memory (i.e., tokens): “As an AI language model, I am based on the GPT-4 architecture, but I don't have a specific token limitation as a standalone entity. However, for practical purposes, platforms and applications using my capabilities may impose their own token limits for input and output.” [100]

According to ChatGPT: “To give you a general idea, GPT-3, my predecessor, had a token limit of 4096 tokens for a single interaction, including both input and output. The token limit for GPT-4 may be similar or slightly higher. Keep in mind that tokens can vary in length, ranging from a single character to a whole word or more. “ [100]

During the diploma work ChatGPT provided short disclaimer: “ChatGPT May 24 Version. ChatGPT may produce inaccurate information about people, places, or facts.”. This research has also excluded all privacy, morality, and security considerations.

7.3 Usage of ChatGPT

In the empirical part of this study, ChatGPT played a crucial role, acting as a key instrument to tackle issues related to agile project difficulties. It can be likened to suggesting the appropriate solution. The system's use in this context is a prominent and significant contribution to the Design Science Research (DSR) concerning prompt patterns. ChatGPT was experimented with various prompt patterns, and the data gathered showcases GenAI's capability in addressing and assisting with varied agile project pain points as prompted. One can say that GenAI in this context is not direct replacement for e.g., project manager to manage agile project pain points but it can help to reduce the manual workload done by experienced project manager.

Furthermore, the creation of this report was greatly aided by the extensive use of ChatGPT (GPT-4) for improving English prose. While ChatGPT has shown its proficiency in delivering in-depth knowledge on project management, its use as a reference has been acknowledged in this document. Much like contemporary text-processing tools like

Microsoft Word, which provides features like a Thesaurus and vocabulary tips, ChatGPT was harnessed for its capability to paraphrase parts of the text.

The focal point of this study has been the utilization of ChatGPT, which has emerged as an indispensable instrument in research across diverse tasks. However, it is crucial to exercise caution while trusting all the generated outputs, considering ChatGPT's status as a Pretrained LLM (Large Language Model).

7.4 Future research opportunities

The realm of GenAI and its uses are swiftly advancing. This research offers a concise glimpse into harnessing AI for project management, with a particular focus on GenAI. As of this document's date, ChatGPT has evolved from the GPT-3 version to the more advanced GPT-4. Research into intelligent project management is just scratching the surface, and there's much more about GenAI's potential yet to be unveiled. This study essentially offers a teaser of what's possible, and the global community is keenly watching the evolution of GenAI systems. Besides ChatGPT, there are other evolving tools and methodologies designed to handle more than just textual data. It's highly likely that various AI tools can be used to assist and augment project management in the future, offering more sophisticated solutions than just typical online search engines.

GenAI offers numerous suggestions for agile project management and can be particularly beneficial in the initial phases when familiarity with agile practices and expertise is limited. However, the extent to which GenAI can assist in projects with highly skilled participants is still unclear. While basic prompt structures appear effective currently, exploring more complex prompt designs and their various combinations presents a captivating area for future investigation.

Although experienced project manager could manage to produce similar findings about managing agile project pain points these GenAI systems such as ChatGPT can provide lots of knowledge of best practices used and experience level knowledge of different projects executed in the history. Probably the training set used in ChatGPT [100] may play big role when applying the latest methods and tools invented during the two years. As stated ChatGPT is a generic LLM model. How about if it could be pre-trained and fine-tuned only for managing pain points in agile project or generally focusing project management only? Also, how the actual tasks of the project manager could be automated by AI remains open.

REFERENCES

[1] "A Guide to the Project Management Body of Knowledge (PMBOK® Guide) – Seventh Edition and The Standard for Project Management (ENGLISH)." https://learning.oreilly.com/library/view/a-guide-to/9781628256642/p01-chapter03.xhtml#sec3.11 (accessed Apr. 03, 2023).

[2] M. (Mark C.) Layton and S. J. Ostermiller, "Agile project management for dummies / by Mark C. Layton and Steven J. Ostermiller," 2017.

[3] "Agile Practice Guide," *Project Management Institute, Inc.*, pp. 57–59, 2017.

[4] R. Hoda and L. K. Murugesan, "Multi-level agile project management challenges: A self-organizing team perspective," *Journal of Systems and Software*, vol. 117, pp. 245–257, Jul. 2016, doi: 10.1016/j.jss.2016.02.049.

[5] J. Tian, Z. Tu, N. Li, T. Su, X. Xu, and Z. Wang, "Intention model based multi-round dialogue strategies for conversational AI bots," *Applied Intelligence*, vol. 52, no. 12, pp. 13916–13940, Sep. 2022, doi: 10.1007/s10489-022-03288-8.

[6] "ChatGPT: Optimizing Language Models for Dialogue." https://openai.com/blog/chatgpt/ (accessed Feb. 02, 2023).

[7] "What is generative AI? | IBM Research Blog." https://research.ibm.com/blog/what-is-generative-AI (accessed May 13, 2023).

[8] "AI Smart Kit: Agile Decision-Making on AI (Abridged Version)." https://web.s.ebscohost.com/ehost/ebookviewer/ebook/ZTAwMHh3d19fMjkxMjA3OF9fQU41?sid=58909a3d-49d0-4dfb-8e4a-d794808d7e26@redis&vid=1&format=EB&lpid=lp_4&rid=0&runquerystringmethod=eBookDownloadundefined_click_bookcheckouts%3acheckBookAvailability%3aready (accessed May 13, 2023).

[9] "How AI Will Transform Project Management." https://hbr.org/2023/02/how-ai-will-transform-project-management (accessed Mar. 17, 2023).

[10] “Agile, Traditional, and Hybrid Approaches to Project Success: Is Hybrid a Poor Second Choice? PMJ ® Practitioner Insights Method Summary”, doi: 10.1177/8756972818823304.

[11] H. Snyder, “Literature review as a research methodology: An overview and guidelines,” *J Bus Res*, vol. 104, pp. 333–339, Nov. 2019, doi: 10.1016/J.JBUSRES.2019.07.039.

[12] “‘Inside Agile Family Software Development Methodologies’ - Google Scholar.” https://scholar-google-com.libproxy.tuni.fi/scholar?as_q=%22Inside+Agile+Family+Software+Development+Methodologies%22&ie=utf8&oe=utf8 (accessed Feb. 09, 2023).

[13] “Manifesto for Agile Software Development.” http://agilemanifesto.org/ (accessed Feb. 05, 2023).

[14] J. Martin, “Agile Project Management with Scrum,” in *Measuring and Improving Performance*, 2009. doi: 10.1201/9781420084191-c2.

[15] B. Boehm and R. Turner, “Management challenges to implementing agile processes in traditional development organizations,” *IEEE Softw*, 2005, doi: 10.1109/MS.2005.129.

[16] M. Perkusich *et al.*, “Intelligent software engineering in the context of agile software development: A systematic literature review,” *Inf Softw Technol*, vol. 119, Mar. 2020, doi: 10.1016/j.infsof.2019.106241.

[17] “inside front cover | Conversational AI.” https://learning.oreilly.com/library/view/conversational-ai/9781617298837/OEBPS/Text/ifc.htm (accessed Feb. 09, 2023).

[18] “What is Scrum? | Scrum.org.” https://www.scrum.org/resources/what-is-scrum (accessed Feb. 05, 2023).

[19] K. Schwaber and M. Beedle, *Agile Software Development with Scrum*. 2001. doi: 10.1109/2.947100.

[20] M. Due Kadenic, K. Koumaditis, and L. Junker-Jensen, “Mastering scrum with a focus on team maturity and key components of scrum,” 2022, doi: 10.1016/j.infsof.2022.107079.

[21] “Extreme Programming | Extreme Programming Installed.” https://learning.oreilly.com/library/view/extreme-programming-installed/0201708426/0201708426_ch01.html (accessed Feb. 06, 2023).

[22] W. Zayat and O. Senvar, “Framework Study for Agile Software Development Via Scrum and Kanban,” *International Journal of Innovation and Technology Management*, vol. 17, no. 4, Jun. 2020, doi: 10.1142/S0219877020300025.

[23] M. Poppendieck and T. Poppendieck, “Implementing Lean Software Development: From Concept to Cash,” *The Addison-Wesley Signature Series*, 2006.

[24] “Lean Software Development: An Agile Toolkit.” https://learning.oreilly.com/library/view/lean-software-development/0321150783/ch01.html (accessed Feb. 06, 2023).

[25] “Interview: Mary and Tom Poppendieck on using Lean for Competitive Advantage.” https://www.infoq.com/interviews/poppendieck-lean-2007/ (accessed Feb. 08, 2023).

[26] “SAFe 5.0 Framework - SAFe Big Picture.” https://www.scaledagileframework.com/ (accessed Sep. 19, 2020).

[27] “2. LeSS - Large-Scale Scrum: More with LeSS.” https://learning.oreilly.com/library/view/large-scale-scrum-more/9780133813142/ch02.html#ch02lev2sec1 (accessed Oct. 10, 2020).

[28] “Disciplined Agile Delivery: A Practitioner’s Guide to Agile Software Delivery in the Enterprise.” https://learning.oreilly.com/library/view/disciplined-agile-delivery/9780132810098/?sso_link=yes&sso_link_from=tampere-university (accessed Feb. 05, 2023).

[29] A. Putta, Ö. Uludağ, M. Paasivaara, and S. L. Hong, “Benefits and Challenges of Adopting SAFe - An Empirical Survey,” *Lecture Notes in Business Information Processing*, vol. 419 LNBIP, pp. 172–187, 2021, doi: 10.1007/978-3-030-78098-2_11.

[30] “Cover Page | SAFe® 4.5 Distilled: Applying the Scaled Agile Framework® for Lean Enterprises, Second Edition.” https://learning.oreilly.com/library/view/safe-r-4-5-distilled/9780135171264/cover.xhtml (accessed Feb. 18, 2023).

[31] "SAFe 5 for Lean Enterprises." https://www.scaledagileframework.com/ (accessed Feb. 18, 2023).

[32] "SAFe Lean-Agile Principles - Scaled Agile Framework." https://scaledagileframework.com/safe-lean-agile-principles/ (accessed Jun. 01, 2023).

[33] "About This E-Book | Large-Scale Scrum: More with LeSS." https://learning.oreilly.com/library/view/large-scale-scrum-more/9780133813142/pref00.html (accessed Feb. 18, 2023).

[34] "Introduction to Disciplined Agile | Disciplined Agile." https://www.pmi.org/disciplined-agile/introduction-to-disciplined-agile (accessed Feb. 06, 2023).

[35] "Chapter 3: Disciplined Agile Delivery in a Nutshell | Introduction to Disciplined Agile Delivery - Second Edition." https://learning.oreilly.com/library/view/introduction-to-disciplined/9781628256543/Chapter03.xhtml (accessed Feb. 05, 2023).

[36] "The PRINCE2 Agile® Practical Implementation Guide – Step-by-step advice for every project type, Second edition." https://learning.oreilly.com/library/view/the-prince2-agile-r/9781787783355/?sso_link=yes&sso_link_from=tampere-university (accessed Jan. 12, 2023).

[37] Sutherland Jeff, "The Scrum@Scale Guide." pp. 4–9, Feb. 2022. Accessed: May 30, 2023. [Online]. Available: https://www.scrumatscale.com/wp-content/uploads/2020/12/official-scrum-at-scale-guide.pdf

[38] "Gartner Says 80 Percent of Today's Project Management Tasks Will Be Eliminated by 2030 as Artificial Intelligence Takes Over." https://www.gartner.com/en/newsroom/press-releases/2019-03-20-gartner-says-80-percent-of-today-s-project-management (accessed Mar. 17, 2023).

[39] V. Prifti, "Optimizing Project Management using Artificial Intelligence," *European Journal of Formal Sciences and Engineering*, vol. 5, no. 1, pp. 30–38, Apr. 2022, doi: 10.26417/667hri67.

[40] M. A. Boschetti, M. Golfarelli, S. Rizzi, and E. Turricchia, "A Lagrangian heuristic for sprint planning in agile software development," 2013, doi: 10.1016/j.cor.2013.09.007.

[41] Y. Sheoraj and R. Kevin Sungkur, “Using AI to develop a framework to prevent employees from missing project deadlines in software projects-case study of a global human capital management (HCM) software company,” 2022, doi: 10.1016/j.advengsoft.2022.103143.

[42] “State of AI Report 2022 - ONLINE - Google Slides.” https://docs.google.com/presentation/d/1WrkeJ9-CjuotTXoa4ZZlB3UP-BXpxe4B3FMs9R9tn34I/edit#slide=id.g164b1bac824_0_2748 (accessed Feb. 25, 2023).

[43] “What is Machine Learning? | IBM.” https://www.ibm.com/topics/machine-learning (accessed Jun. 06, 2023).

[44] “Natural Language Processing in Artificial Intelligence | Brojo Kishore.” https://www-taylorfrancis-com.libproxy.tuni.fi/pdfviewer/ (accessed Feb. 05, 2023).

[45] “Baidu Research.” http://research.baidu.com/Blog/index-view?id=165 (accessed Feb. 09, 2023).

[46] “Generative AI – What is it and How Does it Work?” https://www.nvidia.com/en-us/glossary/data-science/generative-ai/ (accessed Jun. 06, 2023).

[47] “Communicating with Your Customers: The Evolution of Conversational AI | Communicating with Your Customers.” https://learning.oreilly.com/library/view/communicating-with-your/9781492073260/ch01.html#the_evolution_from_basic_chatbots_to_advanced_conv (accessed Feb. 09, 2023).

[48] “ChatGPT AI Model.” https://chat.openai.com/chat (accessed Feb. 06, 2023).

[49] “Google AI updates: Bard and new AI features in Search.” https://blog.google/technology/ai/bard-google-ai-search-updates/ (accessed Feb. 07, 2023).

[50] “China’s Baidu reveals its own generative AI chatbot • The Register.” https://www.theregister.com/2023/02/07/baidu_erniebot_generative_ai_chatbot/ (accessed Feb. 09, 2023).

[51] Y. Shen *et al.*, “ChatGPT and Other Large Language Models Are Double-edged Swords Manuscript Type: Editorial Opinion The work originated from the”.

[52] "2023-Alan-D-Thompson-AI-Bubbles-Rev-7.png (1920×1080)." https://s10251.pcdn.co/wp-content/uploads/2023/03/2023-Alan-D-Thompson-AI-Bubbles-Rev-7.png (accessed Mar. 17, 2023).

[53] OpenAI, "GPT-4 Technical Report," 2023. Accessed: Mar. 15, 2023. [Online]. Available: https://cdn.openai.com/papers/gpt-4.pdf

[54] Kerner Sean Michael, "What is a large language model (LLM)? – TechTarget Definition." https://www.techtarget.com/whatis/definition/large-language-model-LLM (accessed Jun. 05, 2023).

[55] J. Devlin, M.-W. Chang, K. Lee, K. T. Google, and A. I. Language, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", Accessed: Jun. 02, 2023. [Online]. Available: https://github.com/tensorflow/tensor2tensor

[56] Markowitz Dale, "Transformers, Explained: Understand the Model Behind GPT-3, BERT, and T5." https://daleonai.com/transformers-explained (accessed Jun. 05, 2023).

[57] A. Vaswani *et al.*, "Attention Is All You Need," *Adv Neural Inf Process Syst*, vol. 2017-December, pp. 5999–6009, Jun. 2017, Accessed: Jun. 05, 2023. [Online]. Available: https://arxiv.org/abs/1706.03762v5

[58] "How Transformers Work. Transformers are a type of neural… | by Giuliano Giacaglia | Towards Data Science." https://towardsdatascience.com/transformers-141e32e69591 (accessed Jun. 06, 2023).

[59] R. Thoppilan *et al.*, "LaMDA: Language Models for Dialog Applications," Jan. 2022, doi: 10.48550/arxiv.2201.08239.

[60] "Kokeile Bardia, Googlen tekoälykokeilua." https://bard.google.com/ (accessed Jul. 22, 2023).

[61] "The new Bing - Our approach to Responsible AI." Microsoft, 2023. Accessed: Feb. 28, 2023. [Online]. Available: https://blogs.microsoft.com/wp-content/uploads/prod/sites/5/2023/02/The-new-Bing-Our-approach-to-Responsible-AI.pdf

[62] “Does ChatGPT remember what happened earlier in the conversation? | OpenAI Help Center.” https://help.openai.com/en/articles/6787051-does-chatgpt-remember-what-happened-earlier-in-the-conversation (accessed Feb. 25, 2023).

[63] “ChatGPT: What Is It & How Can You Use It?” https://www.searchenginejournal.com/what-is-chatgpt/473664/#close (accessed Feb. 07, 2023).

[64] Montti Roger, “ChatGPT: What Is It & How Can You Use It?” https://www.searchenginejournal.com/what-is-chatgpt/473664/#close (accessed Jun. 06, 2023).

[65] “Introducing ChatGPT.” https://openai.com/blog/chatgpt (accessed Jun. 06, 2023).

[66] “GPT-4 is bigger and better than ChatGPT—but OpenAI won’t say why | MIT Technology Review.” https://www.technologyreview.com/2023/03/14/1069823/gpt-4-is-bigger-and-better-chatgpt-openai/ (accessed May 31, 2023).

[67] “AI chatbots aren’t trustworthy. Could OpenAI, Google or others fix it? - The Washington Post.” https://www.washingtonpost.com/technology/2023/05/30/ai-chatbots-chatgpt-bard-trustworthy/ (accessed May 31, 2023).

[68] “What is GPT AI? - Generative Pre-Trained Transformers Explained - AWS.” https://aws.amazon.com/what-is/gpt/ (accessed Jun. 06, 2023).

[69] “GitHub Copilot · Your AI pair programmer · GitHub.” https://github.com/features/copilot (accessed Jul. 22, 2023).

[70] J. Nuottila, K. Aaltonen, and J. Kujala, “Challenges of adopting agile methods in a public organization,” *International Journal of Information Systems and Project Management*, vol. 4, no. 3, pp. 65–85, Feb. 2022, doi: 10.12821/ijispm040304.

[71] K. Dikert, M. Paasivaara, and C. Lassenius, “The Journal of Systems and Software Challenges and success factors for large-scale agile transformations: A systematic literature review,” *J Syst Softw*, vol. 119, pp. 87–108, 2016, doi: 10.1016/j.jss.2016.06.013.

[72] P. Gregory, L. Barroca, H. Sharp, A. Deshpande, and K. Taylor, “The challenges that challenge: Engaging with agile practitioners’ concerns,” *Inf Softw Technol*, vol. 77, pp. 92–104, 2016, doi: 10.1016/j.infsof.2016.04.006.

[73] J. Patel and Y. Tymchenko, “Making Sense of Resistance to Agile Adoption Making Sense of Resistance to Agile Adoption in Waterfall Organizations: Social Intelligence and Leadership”.

[74] R. Reunamäki and C. F. Fey, “Remote agile: Problems, solutions, and pitfalls to avoid,” *Bus Horiz*, Oct. 2022, doi: 10.1016/J.BUSHOR.2022.10.003.

[75] K. Conboy and N. Carroll, “Implementing Large-Scale Agile Frameworks: Challenges and Recommendations,” *IEEE Software*. 2019. doi: 10.1109/MS.2018.2884865.

[76] R. Hoda and L. K. Murugesan, “Multi-level agile project management challenges: A self-organizing team perspective,” *Journal of Systems and Software*, vol. 117, pp. 245–257, Jul. 2016, doi: 10.1016/J.JSS.2016.02.049.

[77] M. Shameem, R. Ranjan Kumar, C. Kumar, B. Chandra, and A. A. Khan, “Prioritizing challenges of agile process in distributed software development environment using analytic hierarchy process,” 2018, doi: 10.1002/smr.1979.

[78] E. Kula, E. Greuter, A. van Deursen, and G. Gousios, “Factors Affecting On-Time Delivery in Large-Scale Agile Software Development”, doi: 10.1109/TSE.2021.3101192.

[79] J. Sithambaram, M. H. N. B. M. Nasir, and R. Ahmad, “Issues and challenges impacting the successful management of agile-hybrid projects: A grounded theory approach,” *International Journal of Project Management*, vol. 39, no. 5, pp. 474–495, Jul. 2021, doi: 10.1016/j.ijproman.2021.03.002.

[80] I. Hille, “Impact of Remote Working on the Employees During COVID-19 Pandemic-Conclusions From Creative Specialists and Software Developers in the Perspective of the Age of Industry 4.0,” *Management Studies*, vol. 9, no. 6, pp. 439–447, doi: 10.17265/2328-2185/2021.06.002.

[81] M. Paasivaara, B. Behm, C. Lassenius, and M. Hallikainen, “Large-scale agile transformation at Ericsson: a case study,” *Empir Softw Eng*, 2018, doi: 10.1007/s10664-017-9555-8.

[82] K. Vlaanderen, S. Jansen, S. Brinkkemper, and E. Jaspers, “The agile requirements refinery: Applying SCRUM principles to software product management,” 2010, doi: 10.1016/j.infsof.2010.08.004.

[83] T. Kamal, Q. Zhang, and M. A. Akbar, “Toward successful agile requirements change management process in global software development: a client-vendor analysis,” 2019, doi: 10.1049/iet-sen.2019.0128.

[84] N. B. Moe, T. Dingsøyr, and T. Dybå, “Understanding self-organizing teams in agile software development,” *Proceedings of the Australian Software Engineering Conference, ASWEC*, pp. 76–85, 2008, doi: 10.1109/ASWEC.2008.4483195.

[85] H. Barke and L. Prechelt, “Role clarity deficiencies can wreck agile teams,” *PeerJ Comput Sci*, vol. 5, pp. 1–20, 2019, doi: 10.7717/PEERJ-CS.241.

[86] K. Beck *et al.*, “Manifesto for Agile Software Development,” *The Agile Alliance*, 2001.

[87] “Jira | Issue & Project Tracking Software | Atlassian.” https://www.atlassian.com/software/jira (accessed Mar. 07, 2023).

[88] Mr. O’Reilly, “In-Depth Report Engaging people and building processes to accelerate results The Drivers of Agility”.

[89] E. Colonese, “Agile: The human factors as the weakest link in the chain,” *Communications in Computer and Information Science*, vol. 422, pp. 59–73, 2016, doi: 10.1007/978-3-319-27896-4_6/FIGURES/3.

[90] P. J. Guinan, S. Parise, and N. Langowitz, “Creating an innovative digital project team: Levers to enable digital transformation”, doi: 10.1016/j.bushor.2019.07.005.

[91] “IEEE Xplore Full-Text PDF:” https://ieeexplore-ieee-org.libproxy.tuni.fi/stamp/stamp.jsp?tp=&arnumber=1504661 (accessed Mar. 07, 2023).

[92] M. Usman, E. Mendes, F. Weidt, and R. Britto, “Effort Estimation in Agile Software Development: A Systematic Literature Review,” *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*, 2014, doi: 10.1145/2639490.

[93] "IEEE Xplore Full-Text PDF:" https://ieeexplore-ieee-org.libproxy.tuni.fi/stamp/stamp.jsp?tp=&arnumber=9732669&tag=1 (accessed Mar. 01, 2023).

[94] T. O. A. Lehtinen, M. V Mäntylä, J. Vanhanen, J. Itkonen, and C. Lassenius, "Perceived causes of software project failures-An analysis of their relationships," 2014, doi: 10.1016/j.infsof.2014.01.015.

[95] J. vom Brocke, A. Hevner, and A. Maedche, "Introduction to Design Science Research," 2020, pp. 1–13. doi: 10.1007/978-3-030-46781-4_1.

[96] N. Koppenhagen, O. Gaß, and B. Müller, "DESIGN SCIENCE RESEARCH IN ACTION-ANATOMY OF SUCCESS CRITICAL ACTIVITIES FOR RIGOR AND RELEVANCE." Fakultät für Wirtschaftswissenschaften – Institut für Informationswirtschaft und Marketing (IISM), 2012. Accessed: Jul. 20, 2023. [Online]. Available: https://publikationen.bibliothek.kit.edu/1000055012

[97] J. White *et al.*, "A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT," 2023.

[98] "Mastering ChatGPT: How to Craft Effective Prompts (Full Guide + Examples)." https://gptbot.io/master-chatgpt-prompting-techniques-guide/ (accessed Jun. 01, 2023).

[99] "Mastering Generative AI and Prompt Engineering: A Practical Guide for Data Scientists 1.1. Evolution of AI: From rule-based to generative models 1.2. Key generative AI models: RNNs, LSTMs, GPT, and more 1.3. Popular use cases for generative AI".

[100] "ChatGPT," 2023. https://chat.openai.com/chat?model=gpt-4 (accessed Mar. 19, 2023).

[101] "OpenAI API." https://platform.openai.com/tokenizer (accessed Jun. 07, 2023).

[102] R. T. Kreutzer, T. Neugebauer, and A. Pattloch, "Change Management: Shaping Change Successfully," 2018, pp. 197–218. doi: 10.1007/978-3-662-56548-3_3.

[103] "Managing Software Requirements the Agile Way | Managing Software Requirements the Agile Way." https://learning.oreilly.com/library/view/managing-software-requirements/9781800206465/B16234_FM_Final_NM.xhtml (accessed Feb. 15, 2023).

[104] T. Raharjo and B. Purwandari, "Agile project management challenges and mapping solutions: A systematic literature review," *ACM International Conference Proceeding Series*, pp. 123–129, Jan. 2020, doi: 10.1145/3378936.3378949.

[105] D. Koutsikouri, S. Madsen, and N. B. Lindström, "Agile Transformation: How Employees Experience and Cope with Transformative Change," Springer, Cham, 2020, pp. 155–163. doi: 10.1007/978-3-030-58858-8_16.

APPENDIX A: PROMPT TEST ROUND 1

Model: GPT-4

Requirements creation prompt:

$ As a requirement engineer your task is to create a requirement specification. The requirement specification must fulfill the following constraints: Internet banking system which can handle typical banking transactions to be possible to browse account and make payments. Ask the necessary questions needed to fully understand the constraints. Based on my answers create requirement specification. Explain your understanding and construct your proposal to me representing the project manager. Provide your output in Table format.

ChatGPT

Sure, I will need a few more details to make sure we're on the same page. Here are the questions that I'd need answers to:

1. Are there any specific types of banking transactions that the system must support (e.g., wire transfers, automatic bill payments, mobile check deposit)?
2. What types of accounts should users be able to browse (e.g., checking, savings, credit card, loan)?
3. Who are the expected users of this system (e.g., retail customers, business customers, bank employees)?
4. What levels of access should different users have?
5. Are there specific platforms the system should be available on (e.g., Web, Mobile App)?
6. What types of payment methods should users be able to manage (e.g., direct debit, credit/debit card, mobile wallets)?
7. Should the system support multi-currency transactions?
8. What are the security requirements for the system (e.g., two-factor authentication, biometric security)?

9. What are the requirements for regulatory compliance (e.g., data privacy, transaction reporting)?
10. Should the system include features for financial management or advice (e.g., spending analysis, budgeting tools, investment advice)?
11. Are there any particular interfaces with other systems (e.g., other banks for interbank transactions, credit rating agencies)?
12. Are there any expected non-functional requirements, such as availability, performance, or disaster recovery?

Based on the answers to these questions, I would then construct a detailed requirement specification. The Table might look like the following example, which I'll fill out based on some common assumptions:

Requirement ID	**Requirement Description**	**Requirement Type**	**Source**
R1	The system must support typical banking transactions including deposits, withdrawals, transfers, and bill payments.	Functional	Stakeholder Interviews
R2	Users must be able to browse checking, savings, and loan account information.	Functional	Stakeholder Interviews
R3	The system must support retail customers and bank employees as users.	Functional	Stakeholder Interviews
R4	The system must provide different levels of access, with retail customers able to access only their own accounts, and bank employees able to access multiple accounts based on their role.	Functional	Stakeholder Interviews

Requirement ID	Requirement Description	Requirement Type	Source
R5	The system must be available as a web application and mobile app.	Non-Functional	Stakeholder Interviews
R6	The system must support direct debit and credit/debit card payments.	Functional	Stakeholder Interviews
R7	The system must support multi-currency transactions.	Functional	Stakeholder Interviews
R8	The system must support two-factor authentication for user logins.	Functional	Stakeholder Interviews
R9	The system must comply with data privacy regulations and transaction reporting requirements.	Non-Functional	Regulatory Compliance
R10	The system should provide financial management features, including spending analysis and budgeting tools.	Functional	Stakeholder Interviews
R11	The system should interface with other banks for interbank transactions and credit rating agencies for credit checks.	Functional	Stakeholder Interviews

Requirement ID	Requirement Description	Requirement Type	Source
R12	The system should have an availability of 99.9% and a disaster recovery plan in place.	Non-Functional	Business Continuity Plan

Please provide the information for the questions I asked above so that I can create a more specific requirement specification for your project.

$ Users are retail customers and end user customers. Account can be used for checking, savings and credit cards transactions. Payment types should be related to ecommerce so credit cards and money transfer. Recurring payments should be provided. Security follows normal banking security. MFA should be supported through mobile app. The system is integrated with ecommerce shop. GDPR regulations apply. User need to have bank account statements report of the transactions made. Customer support should have chat and normal telephone. In addition FAQ is good to have.

ChatGPT

Thanks for the information! Based on your responses, here is a revised Table of requirements:

Requirement ID	Requirement Description	Requirement Type	Source
R1	The system must support typical banking transactions including deposits, withdrawals, transfers, and bill payments.	Functional	Stakeholder Interviews

Requirement ID	Requirement Description	Requirement Type	Source
R2	Users must be able to browse checking, savings, and credit card account information.	Functional	Stakeholder Interviews
R3	The system must support retail customers and end-user customers.	Functional	Stakeholder Interviews
R4	The system must provide different levels of access, with retail and end-user customers able to access only their own accounts.	Functional	Stakeholder Interviews
R5	The system must support credit card payments and money transfers.	Functional	Stakeholder Interviews
R6	The system must support recurring payments.	Functional	Stakeholder Interviews
R7	The system must provide banking-level security with multi-factor authentication through a mobile app.	Functional	Stakeholder Interviews
R8	The system must be integrated with an ecommerce shop.	Functional	Stakeholder Interviews
R9	The system must comply with GDPR regulations.	Non-Functional	Regulatory Compliance
R10	Users should be able to access bank account statements reporting their transactions.	Functional	Stakeholder Interviews

Requirement ID	Requirement Description	Requirement Type	Source
R11	Customer support must be available through chat and telephone.	Functional	Stakeholder Interviews
R12	A FAQ section should be available to users.	Functional	Stakeholder Interviews
R13	The system must be available as a web application and mobile app.	Non-Functional	Stakeholder Interviews
R14	The system should have an availability of 99.9% and a disaster recovery plan in place.	Non-Functional	Business Continuity Plan

This Table outlines the system's main functional and non-functional requirements, allowing the developers to understand what they need to build. Additionally, it outlines sources of the requirements, which can be useful for reference and further consultation.

$ Can you prioritize requirements in the Table?

ChatGPT

Sure, I can add a priority column. For the priority, let's use the MoSCoW method where:

- **M (Must have)**: These are critical requirements that the system cannot function without.
- **S (Should have)**: Important but not vital requirements, and can be delayed if necessary.
- **C (Could have)**: These are nice to have requirements that could improve user experience or system performance but are not necessary.
- **W (Won't have)**: These are least-critical or unnecessary right now, but could be considered in the future.

Here is the revised Table with the priority column:

Requirement ID	**Requirement Description**	**Requirement Type**	**Source**	**Priority**
R1	The system must support typical banking transactions including deposits, withdrawals, transfers, and bill payments.	Functional	Stakeholder Interviews	Must Have
R2	Users must be able to browse checking, savings, and credit card account information.	Functional	Stakeholder Interviews	Must Have
R3	The system must support retail customers and end-user customers.	Functional	Stakeholder Interviews	Must Have
R4	The system must provide different levels of access, with retail and end-user customers able to access only their own accounts.	Functional	Stakeholder Interviews	Must Have
R5	The system must support credit card payments and money transfers.	Functional	Stakeholder Interviews	Must Have
R6	The system must support recurring payments.	Functional	Stakeholder Interviews	Should Have
R7	The system must provide banking-level security with multi-factor authentication through a mobile app.	Functional	Stakeholder Interviews	Must Have
R8	The system must be integrated with an ecommerce shop.	Functional	Stakeholder Interviews	Must Have

Requirement ID	Requirement Description	Requirement Type	Source	Priority
R9	The system must comply with GDPR regulations.	Non-Functional	Regulatory Compliance	Must Have
R10	Users should be able to access bank account statements reporting their transactions.	Functional	Stakeholder Interviews	Should Have
R11	Customer support must be available through chat and telephone.	Functional	Stakeholder Interviews	Should Have
R12	A FAQ section should be available to users.	Functional	Stakeholder Interviews	Could Have
R13	The system must be available as a web application and mobile app.	Non-Functional	Stakeholder Interviews	Must Have
R14	The system should have an availability of 99.9% and a disaster recovery plan in place.	Non-Functional	Business Continuity Plan	Should Have

This Table will help in understanding which requirements to focus on first during the development process and which ones can be pushed back if required.

Model: GPT-4

Steering group prompt:

$ As a project’s steering group member your task is to fully understand agile principles, business objectives and give feedback about the project. The project has the following objectives: We are implanting an ERP integration system to customer using SAP product. Integration is done in agile way and we expect to release one integration in every two weeks. At the moment we do not get customer to participate demos and would need their feedback. Project cannot start final testing before customer testing. Customer does not have time for the testing but is expecting the release soon. Ask the necessary questions to fully understand the objectives to be able to understand the project status and way of working. Explain your understanding and provide necessary actions for me as the project manager. Provide output in memo format.

ChatGPT

MEMO

To: Project Manager

From: ChatGPT, Project Steering Group Member

Subject: ERP Integration Project Understanding and Recommendations

Date: 17th May, 2023

Dear Project Manager,

I understand that our current endeavor involves agile development and deployment of an ERP integration system using SAP products for our customer. We are aiming for a release cycle of one integration every two weeks. However, we are facing challenges in obtaining customer feedback during demos and in the final testing phase due to their time constraints.

To further understand the situation and propose effective solutions, I need additional information on the following:

6. **Customer Engagement:** What are the current methods of customer engagement and how frequently are we reaching out to them for demos and feedback?

7. **Customer Representative:** Is there a dedicated customer representative involved in the project who can spare time for regular feedback and testing?
8. **Testing:** What is the nature and extent of testing required from the customer before we can move to final testing?
9. **Release Expectations:** What are the specific timeframes when the customer is expecting the releases?
10. **Communication and Collaboration Tools:** What tools are we currently using to facilitate communication and collaboration between our team and the customer?

Understanding these specifics will help us form a more effective strategy for customer engagement and project execution.

Based on the current situation, here are some initial recommendations:

1. **Identify a Customer Representative:** It would be beneficial to have a dedicated customer representative who can actively participate in the project. This representative would be responsible for providing feedback, attending demos, and participating in testing.
2. **Flexible Demo and Feedback Sessions:** To accommodate the customer's time constraints, we could offer flexible options for demo and feedback sessions, such as recorded demos that can be watched at their convenience.
3. **Effective Communication Channels:** Leveraging efficient

$ We have customer representative but she is not so active, however we communicate on weekly basis. We would like to know customer testing that everything is ok and we can proceed to final testing. Customer is expecting the release based on the date given on the contract. With customer we use Teams on weekly basis to share also documentation.

ChatGPT

MEMO

To: Project Manager

From: ChatGPT, Project Steering Group Member

Subject: Revised ERP Integration Project Recommendations

Date: 17th May, 2023

Dear Project Manager,

Thank you for the additional information. It's clear that we do have some customer involvement, though the customer representative's participation is somewhat limited. We maintain weekly communication via Microsoft Teams and the customer's primary expectation is to meet the delivery dates specified in the contract. They are also required to validate the product before we proceed to final testing.

Given these details, I suggest the following actions:

1. **Increase Engagement with Customer Representative:** Aim to increase the active involvement of the customer representative by making meetings more efficient and engaging. Clearly outline the meeting agendas in advance, ask for their input on specific issues, and emphasize the importance of their role in the project's success.
2. **Provide Clear Guidelines for Testing:** The customer may be unsure about the extent of testing required from their end. Provide a clear and concise testing guide, outlining exactly what is expected from them and how they can perform these tests. This could potentially save time and expedite the testing process.
3. **Use Teams More Effectively:** Make the most of Microsoft Teams by using it not only for meetings and documentation sharing but also for continuous collaboration. Consider creating a dedicated channel for queries and discussions related to testing and feedback.
4. **Frequent Updates and Transparency:** Keep the customer representative updated about the project status and any potential delays or issues that might impact the delivery date. Transparency can foster trust and more active involvement.

5. **Leverage Recorded Demos:** If live demos aren't feasible due to time constraints, consider providing recorded demos that the customer representative can view and review at her convenience.
6. **Escalation Pathway:** If the customer representative continues to be less active than needed, establish an escalation pathway. Discuss the issue with higher management in the customer's organization if necessary.

It's essential to remember that the customer's involvement in the agile process is crucial for the project's success. We must make every effort to ensure their active participation.

Best Regards,

ChatGPT Project Steering Group Member

Model: GPT-4

Role clarification pattern

$ As an engineering manager your task is to specify clear roles for the team. The project has the following constraints: software is created using C++ and publicly known open source components. Only one person knows how to program in C++ but the rest 4 persons have experience only in Python and Javascript. Team needs to deliver C++ application in Linux environment in the next four months. Ask all necessary questions to fully understand the project and be able guide team to create necessary role specifications. Explain your under-standing and provide solution to me representing the management. Provide output in Table format.

ChatGPT

Sure, I'd first need to understand a few more details about the project to make the role specifications as accurate as possible. Here are the questions:

1. What is the nature and scope of the project? For example, are we developing a web application, a desktop application, a mobile application, a server-side application, or something else?

2. What open source components are we planning to use in the project?
3. Will there be a need for database or server-side interactions in the project?
4. Is there any requirement for web-based user interfaces or front-end development?
5. How complex is the project? Is it a simple application with basic functionality, or a more complex project that will require significant architectural planning and testing?
6. What is the skill level of the Python and Javascript programmers? Are they beginners, intermediate, or advanced?
7. What resources do we have for training the Python/Javascript developers in C++? How much time can be allocated for training?
8. Do we have or need DevOps skills in the team to manage the Linux environment?
9. What is the current state of the project? Is it starting from scratch or is there already some work done?
10. How familiar is the team with software development methodologies (like Agile, Scrum, Kanban etc.)?

Given these details, I will be able to provide a comprehensive solution. After I understand these aspects, I will divide the team into roles such as C++ Developer, Python/Javascript Developer, Tester, DevOps, etc. according to the project needs and the skill set of the team members.

The roles could then be something like this:

- C++ Developer: Responsible for major C++ coding tasks and helping other team members understand C++ code.
- Python/Javascript Developers: Depending on the project, they could be responsible for tasks such as scripting, testing, or developing front-end components. They could also be trained to contribute to C++ development.
- Tester: Responsible for defining and executing test cases, and validating that the software meets the requirements.
- DevOps: If required, responsible for setting up and managing the Linux environment, deployment pipeline, etc.

Once I understand more about the project, I will be able to provide a more detailed Table of roles and responsibilities.

$ Project needs to deliver C++ application in embedded device and transfer the data to backend. It should collect IoT data and move that to the backend for further processing. We use existing cloud based backend but IoT device as Atmel based 32 bit processor and necessary hardware. We would like to utilized existing sw designers also in C++ development. We collaborate through github using its features. Linux is ubuntu based. Testing is done fully manually as we don't have suiTable tools for testing C++ applications and the application is simple. We plan to make some error updates but otherwise maintenance is approx two times in the year."

ChatGPT

Based on the additional details, I suggest the following roles and responsibilities for your team members:

Role	Responsibilities	Member Skills
C++ Lead Developer	Responsible for C++ development in the embedded device, guiding the team on C++ concepts and best practices, interfacing with the IoT device hardware, and integrating with the backend. Also assists in training the Python/Javascript developers in C++.	C++
Python/Javascript Developers	Python/Javascript developers should learn C++ with the assistance of the Lead Developer and online resources. They can contribute to C++ development tasks and also help in scripting, and data processing related tasks. They also can help with error update coding.	Python, Javascript
Manual Tester	Responsible for defining and executing test cases, testing data collection and transmission, validating software requirements, and performing error updates.	Python, Javascript

Role	Responsibilities	Member Skills
DevOps Engineer	Manages the Linux environment (Ubuntu), version control systems (GitHub), cloud operations, and the deployment pipeline. Also responsible for device setup and connectivity.	Linux, Cloud Systems, GitHub
Project Manager	Manages the entire project, resolves any blockers the team faces, ensures timely delivery, and communicates with stakeholders.	Project Management, Knowledge of Agile Methodology

I recommend the team to work in an Agile Scrum methodology given the complexity and novelty of the project. This will allow the team to iterate quickly, handle changes effectively, and continuously improve. The team members should participate in Agile rituals like daily standup, sprint planning, sprint review, and sprint retrospective to ensure everyone is on the same page.

In order to maximize the team’s efficiency, the Python/Javascript developers should begin learning C++ as soon as possible, with the assistance of the C++ lead and online resources. Given the simplicity of the application, it is feasible for them to contribute to C++ development tasks within a few weeks.

For the manual testing, since the team does not have suiTable tools for testing C++, it is crucial for the tester to understand the application deeply and create a solid test plan covering all possible cases.

Regarding the DevOps role, since the application runs in a Linux environment and uses GitHub, this person should have experience with Linux and version control systems. This role is also important for managing the deployment to the embedded devices and ensuring the connectivity and data transmission to the backend.

Finally, a project manager role is necessary to oversee the project, ensure timelines are met, coordinate among the team, and communicate with the stakeholders. The project manager can also help the team to embrace Agile methodology and assist in planning, reviews, and retrospectives.

The specifics of these roles might need to be adjusted based on the specifics of your team and your project, but this should give you a good starting point.

Model: GPT-4

Redundancy analysis prompt:

$ As a resource manager your task is to analyze redundancy in agile team skills. The project team has the following persons: one Java back-end programmer, test automation developer, two JavaScript front-end programmers and DevOps specialist. Responsibilities of the team are: create full-stack application for netbanking system, UI is based on javascript and backend is done using Java. Ask the necessary questions to fully understand the task and create analysis of redundancy gaps in the team. Explain your understanding and provide solution to me representing the project manager. Provide output in Table.

ChatGPT

Sure, to understand the skills and potential redundancy in this agile team, I would need to ask the following questions:

1. What is the level of cross-skills of each team member?
2. Are any of the team members experienced or skilled in multiple roles?
3. What are the levels of proficiency in each team member's skill sets?
4. What specific responsibilities does each team member have in their respective roles?
5. How complex is the netbanking system the team is developing?
6. What is the estimated timeline for the project?
7. Are there any unique responsibilities or tasks that only specific team members can handle?

8. Do the team members have experience in working in agile teams and are they familiar with agile practices like pair programming, collective code ownership, etc.?
9. Do the team members have knowledge and skills in other relevant technologies that the project may require?

Based on these questions, we can analyze the redundancies and gaps in the team.

Analysis

Let's consider for our analysis that all the team members are proficient in their respective roles, they are experienced in agile practices, they have cross-skills but to a limited extent and there are no unique responsibilities that only specific team members can handle.

Given this, the redundancy analysis of the team would look something like this:

Role	Redundancy	Potential Gaps	Solution
Java Backend	Low	High proficiency in Java	Hire an additional Java developer or upskill the existing team members in Java
Test Automation	Low	High proficiency in Test Automation	Upskill the existing team members in test automation
JavaScript Front-end	Moderate	High proficiency in Javascript	No additional resources needed; however, upskill in JS could be beneficial
DevOps	Low	High proficiency in DevOps	Upskill the existing team members in DevOps or consider hiring an additional DevOps specialist

From this Table, we can see that there's limited redundancy in Java back-end programming, test automation, and DevOps, implying that there's a risk if the team member responsible for these roles is absent or leaves the project. The JavaScript front-end has a moderate level of redundancy because there are two programmers for this role.

To mitigate the risk of low redundancy, cross-skilling or upskilling of team members could be considered. Cross-skilling will help the team members to handle the tasks of their peers in their absence. Upskilling will enhance the skills of team members in their respective roles.

In the case where cross-skilling or upskilling is not feasible or enough, hiring additional team members may be necessary. Please note that hiring decisions should be based on the complexity of the system, the estimated timeline of the project, and the capability of the existing team.

Furthermore, the team should continue to foster a culture of collective code ownership and pair programming to promote knowledge sharing and reduce the risk of having only one person familiar with particular parts of the project.

$ Team members are doing their own job and have not been helping in tasks having unknown skills. Two senior developers have done in the past versatile programming with different programming languages. Generally the knowledge of programming is quite good. It would be good that persons could do more cross-functional development. Netbank is quite complex as it needs UI and backend knowledge in security and banking domain. Project is ongoing and releases are around four times per year. Only one person can do the backend at the moment. You have good questions and I cannot give answer for each.

ChatGPT

Based on the new information, let's reassess the redundancy analysis:

Role	Redundancy	Potential Gaps	Solution
Java Back-end	Low	Java, Security, Banking Domain	Hire an additional Java developer with knowledge in security and banking domain, or upskill one of the senior developers
Test Automation	Low	High proficiency in Test Automation	Upskill one of the senior developers in test automation or consider hiring an additional Test Automation specialist
JavaScript Front-end	Moderate	High proficiency in JavaScript, UI	No additional resources needed; however, upskilling in UI design and JavaScript could be beneficial
DevOps	Low	High proficiency in DevOps	Upskill one of the senior developers in DevOps or consider hiring an additional DevOps specialist

Given the complexity of the netbanking system, it is necessary to ensure that there is enough redundancy in crucial roles such as Java back-end development, especially considering the knowledge requirements in the security and banking domain. Since only one person can currently handle the backend, this is a significant risk to project continuity.

The solution would be to either upskill one of the senior developers in Java and relevant domain knowledge, or hire an additional Java developer with this expertise.

For the Test Automation and DevOps roles, it would be beneficial to upskill one of the senior developers in these areas to provide redundancy. If this is not feasible, hiring additional resources should be considered.

The team has a good redundancy in JavaScript front-end development. However, considering the complexity of the UI, further upskilling could be beneficial to ensure the quality of the work.

Finally, I'd recommend setting a clear expectation of cross-functional development within the team. This involves individuals moving out of their comfort zones to learn new skills and contribute across different areas of the project. This not only builds redundancy but also increases team resilience and adaptability. Implementing practices such as pair programming could help facilitate this cross-skilling, where team members work together on tasks and share knowledge and skills.

Model: GPT-4

Agile process coaching prompt:

$ As an agile coach your task is to coach agile processes. The project has the following objectives: build a web application to be ready in autumn market. We have backlog full of features. Some of the features are quite abstract. Job needs to be done in two weeks iterations and we should be able to test and release the software as often as possible. Project team is five persons with senior and junior designers not familiar of Scrum or agile ways of working. Ask the necessary questions to fully understand the objectives and create agile coaching plan to the team. Explain your understanding and show me the plan representing the project manager. Provide output in action plan format.

ChatGPT

Sure, let's get started by asking a few clarifying questions:

1. What roles do the five team members have? Are there dedicated developers, testers, or product owners?
2. Are there existing processes in place that the team follows, or are we starting from scratch?
3. What are the communication practices currently being used? Do they use any tools for communication or project management?
4. Can you clarify the scope of the application? Are there any technical specifications or constraints we should be aware of?
5. How familiar are the team members with the concept of Agile? Do they have any training or past experience with this methodology?

6. What is the timeline for the project? When is the expected launch date?
7. Who are the key stakeholders, and what are their expectations?

Understanding these aspects will help me create a more tailored Agile coaching plan.

Here's a draft of the Agile Coaching Plan considering the above details:

1. Agile Introduction and Training:

- Objective: To provide a fundamental understanding of Agile methodologies, focusing on Scrum since we're planning for two-week iterations.
- Action: Organize a couple of training sessions about Agile and Scrum, covering its principles, benefits, roles, ceremonies, and artifacts. Include real-world examples to illustrate the concepts better.

2. Team Roles Definition:

- Objective: Define the roles of Scrum Master, Product Owner, and Scrum Team in the context of your team.
- Action: Assign roles to team members based on their skills, interests, and the project requirements. Ensure every team member is clear on their roles and responsibilities.

3. Backlog Refinement:

- Objective: Understand the product backlog, break down large/abstract features into manageable user stories.
- Action: Arrange a series of backlog refinement sessions with the team and stakeholders to understand the product vision and break it down into smaller user stories.

4. Sprint Planning:

- Objective: Plan the work for the next two weeks.
- Action: Facilitate Sprint planning sessions, ensuring the team understands what they're committing to, and the Definition of Done.

5. Agile Ceremonies Establishment:

- Objective: Establish a regular cadence of Sprint Review, Sprint Retrospective, and Daily Stand-up meetings.
- Action: Schedule and facilitate these meetings, ensuring their purpose and format is understood and respected by the team.

6. Agile Tools Introduction:

- Objective: Ensure the team is comforTable with using project management and communication tools.
- Action: Introduce tools like JIRA, Trello, or Asana for project management, and Slack or Microsoft Teams for communication. Provide necessary training.

7. Agile Metrics and Reporting:

- Objective: Track the progress of the project and ensure visibility for stakeholders.
- Action: Introduce the concept of Agile metrics (like Velocity, Burndown charts) and how they'll be used to track and report progress.

8. Continuous Improvement:

- Objective: Improve the process and adapt to changes as the project progresses.
- Action: Use Retrospective meetings to identify what's working and what's not, then implement changes in the next Sprint.

This plan is iterative and will be adjusted as the project progresses and the team gets more comforTable with Agile.

$ All team members are dedicated to their function. Team has tried to use Scrum but with no good luck. As being mostly remotely team has been used Teams with adhoc meetings and some daily startup meetings. This is important project for the company as it is the first web application to the market. Time line for project is the next four months. We would like to release after each sprint but the sprint working is new. So we don't have experience on Scrum practicalities. The only challenge in our team is the level of knowledge.

ChatGPT

Thanks for your detailed responses. Given the additional information you've provided, here's an updated Agile Coaching Plan:

1. Agile/Scrum Refresher:

- Objective: Reinforce the fundamentals of Agile and Scrum, emphasizing practical usage and overcoming past challenges.
- Action: Deliver a comprehensive training session covering Scrum framework's roles, ceremonies, and artifacts, focusing on real-life implementation and how it can address previous challenges.

2. Roles Definition and Responsibilities:

- Objective: Clarify the roles of Scrum Master, Product Owner, and Scrum Team, considering the previous experience.
- Action: Define and reinforce the responsibilities of each role in Scrum. Ensure that each member knows their part and how to collaborate effectively.

3. Sprint Planning and Backlog Refinement:

- Objective: Understand the product backlog, break down large/abstract features into manageable user stories.
- Action: Host backlog grooming sessions, enabling the team to break down and estimate the tasks better. Improve the team's skill at planning and estimating sprints.

4. Tools Training and Efficient Usage:

- Objective: Boost proficiency with project management and communication tools like Microsoft Teams and other Agile tools.

- Action: Provide training sessions focusing on the optimal use of the tools to manage the backlog, sprints, and progress tracking.

5. Regular Agile Ceremonies:

- Objective: Install a regular cadence of Scrum events, including Sprint Planning, Daily Scrum, Sprint Review, and Sprint Retrospective.
- Action: Schedule these events and stick to the timeTable strictly. Ensure that the purpose of each event is clear and it contributes to the sprint's goal.

6. Building Release Cycle:

- Objective: Establish a process for regular releases after each sprint, involving the necessary testing and review processes.
- Action: Collaborate with the development and QA teams to create a release cycle that fits into the two-week sprint structure.

7. Agile Metrics and Reporting:

- Objective: Establish Agile metrics and progress tracking to ensure visibility for stakeholders.
- Action: Teach the team about key Agile metrics, including burndown charts, velocity, and sprint goal success rates. Share regular reports with stakeholders.

8. Cultivating a Learning Culture:

- Objective: Encourage continuous learning and improvement within the team.
- Action: Use Sprint Retrospective meetings to discuss what worked and what didn't, identify learning opportunities, and plan improvements for the next sprint.

9. Stakeholder Management:

- Objective: Ensure clear and open communication between the team and stakeholders.
- Action: Establish a routine of regular updates to stakeholders, invite them to Sprint Review sessions, and encourage their feedback.

This plan aims to not just introduce Agile practices but to also ensure that the team is able to implement them effectively in their work. The focus is on practical application, continuous improvement, and addressing the challenges you've experienced before. The aim is to have a self-sufficient, high-performing Agile team by the end of the four-month period.

Model: GPT-4

Adaptability coaching prompt:

$ As an agile coach your task is to coach team to adapt and motivate in occurring changes. The project has the following objectives: to deliver ERP system integration to new vendor system, vendor is changing their way of working constantly, team has used to work in two weeks sprints, but the customer is working ad hoc and does not respect sprints. Ask the necessary questions to fully understand the objectives and create a plan how to adapt to the changes. Explain your understanding and provide the actions to me representing the project manager. Provide output in Table.

ChatGPT

I understand the challenges that the team is facing - dealing with a vendor that changes its approach constantly, a customer working in an ad hoc fashion, and the disruption to your usual two-week sprint cadence. As an Agile coach, it's important to understand the context deeply to create an effective plan. Here are a few questions I would ask to get a better understanding:

1. What are the specific changes that the vendor is making, and how often do they occur?
2. What type of ERP system are we integrating? Do we have any prior experience with this type of system?
3. What are the communication protocols between our team and the vendor? Are we notified well in advance of any changes?
4. How does the customer's ad hoc working style disrupt our sprint schedule?

5. What is the customer's understanding of our agile process? Have they had any exposure or education on Agile/Scrum methodologies?
6. What challenges, besides the disrupted sprints, has the team been facing due to the ad hoc nature of the customer's work?
7. Do we have any flexibility in aligning our sprints with the customer's work style, or vice versa?
8. Is the team's motivation impacted only by these changes, or are there other factors affecting it?

Based on the information gathered, here is a preliminary action plan that we can adapt and refine based on your responses:

Understanding	**Actions**
Constant changes from the vendor	Establish a clear communication protocol with the vendor. Request advance notice for any changes. Increase the frequency of sync up meetings with the vendor, if necessary.
ERP system integration	Ensure that the team is trained in the ERP system, or have access to necessary resources. Look for opportunities for continuous learning.
Customer's ad hoc working style	Engage the customer in a discussion about how the team works, and the benefits of sticking to a sprint schedule. If flexible, experiment with modifying the sprint duration or adopting a more Kanban-like approach.
Two-week sprint disruption	Look into adopting a more flexible agile framework like ScrumBan or Kanban, which can accommodate a less predicTable workload.

Understanding	Actions
Team motivation	Run retrospectives to understand the team's concerns. Implement changes based on feedback. Encourage team bonding and promote a culture of open communication. Celebrate small wins and make sure the team feels valued.

It's important to remember that change is a part of every project, and adapting to change is a key aspect of Agile. We need to ensure the team understands this and is comforTable with the idea of adapting their work styles as required while staying motivated and productive.

$ Vendor changes happens quite ad hoc but typically in week or two. ERP used is SAP and persons have some knowledge of that as end users. We have weekly meetings over Teams with the vendor. Customer is not aware of vendor schedule as there is not direct channel between them. Customer is expecting releases on time independently on agile system used. We stick in the own release cycle but there is pressure to change that towards customer's schedule. Team is not motivated when lots of unplanned changes are coming.

ChatGPT

Thank you for providing the additional information. I understand your concerns and see the challenges the team is facing with the frequent changes from the vendor, the customer's demand for on-time delivery, and the impact on the team's motivation. Based on your responses, I am refining the action plan as follows:

Understanding	Actions
Vendor changes ad hoc, typically every week or two	Continue with the weekly meetings and encourage the vendor to share any planned changes as early as possible. Develop a strategy to accommodate these changes within your sprint cycle or consider adopting a more flexible approach to the sprints.

Understanding	Actions
Team has some knowledge of SAP as end users	Organize training sessions for the team to upgrade their skills from end users to developers or consultants as needed. Connect them with resources or experts to assist with technical issues.
Customer is not aware of vendor schedule, expects timely releases	Establish a transparent communication channel with the customer, keeping them informed about the project progress and any changes from the vendor side. Explain the possible impact of vendor changes on the release schedule and negotiate realistic deadlines.
Pressure to change release cycle to match customer's schedule	Discuss the feasibility of this with the team. If it's causing stress or overwork, negotiate with the customer to find a balance between their demands and the team's capacity.
Team is demotivated due to frequent unplanned changes	Use retrospectives to give the team a space to voice their concerns. Show empathy and work collaboratively to find solutions. Acknowledge their efforts in managing changes and reassure them of their value to the project. Implement strategies to boost morale like peer recognition, team building activities, etc.

Agile is about delivering value while responding to changes. It's crucial to build an environment where the team feels comforTable in addressing the changes. Aligning the customer, the vendor, and the team might require negotiation and flexibility, but it will lead to a more efficient and motivated team.

Model: GPT-4

Change resistance prompt:

$ As an agile coach your task is to improve team way of working to overcome change resistance. The project team has the following challenges: team has been working in water-fall method over five years, scrum or agile way of working is new to the team, senior developers would like to continue the work as they have used to without dailies and retrospectives. Ask the necessary questions to fully understand the challenges and provide plan how to overcome change resistance. Explain your understanding and thinking to me representing the scrum master in the team. Provide output in action plan format.

ChatGPT

Sure, in order to fully understand the challenges, I would need to ask some questions first:

1. What specific issues or challenges is the team facing that have motivated this move towards Agile methodology? Is there any particular reason for the change?
2. How receptive is the rest of the team, aside from the senior developers, to this change? Are there any who are interested or have some experience with Agile?
3. Is there any support from the higher management for this transition? What resources, like training or coaching, are they willing to provide?
4. Have there been any previous attempts at Agile transformation, and if so, what were the outcomes and what lessons were learned?
5. How do the senior developers feel about change in general, and how do they handle it? What are their specific concerns about moving to Agile?
6. How does the team currently handle project management, and what tools do they use? Are there any elements of their existing system that could be used in a Scrum setting?

Based on the responses to these questions, I would create an action plan to address the resistance to change and transition to Agile. It could look something like this:

Action Plan

Phase 1: Diagnosis and Engagement

1. Conduct a workshop with the team to understand their concerns, expectations, and ideas about the transition.
2. Interview key stakeholders to gain insights into the organizational culture and readiness for change.

3. Analyze the team's current processes to identify areas of strength and improvement, and to see how they align with Agile principles.

Phase 2: Knowledge Sharing and Education

1. Initiate Agile and Scrum training sessions for the team. Start with the basics and gradually introduce more advanced topics.
2. Encourage the team to learn from other teams or organizations that have successfully transitioned to Agile.
3. Create an environment where learning and asking questions is encouraged.

Phase 3: Gradual Implementation

1. Start implementing Agile practices in a controlled environment, like a single small project or a part of a larger project.
2. Encourage the team to hold daily stand-up meetings, even if they're brief at first. Gradually introduce other Scrum ceremonies like retrospectives and sprint planning.
3. Make use of Agile project management tools, and train the team to use them effectively.

Phase 4: Coaching and Support

1. As an Agile coach, provide ongoing support to the team during the transition.
2. Support the Scrum Master in enforcing Scrum principles, handling team dynamics, and managing potential conflict.
3. Encourage the senior developers to take on leadership roles within the Agile framework, such as becoming product owners or Scrum Masters.

Phase 5: Evaluation and Improvement

1. Regularly review the team's progress towards Agile adoption. Use feedback from the team and data from the projects to evaluate success.
2. Celebrate the team's achievements in the transition to Agile to maintain motivation.

3. Continuously improve the process based on feedback from the team and results from the projects.

My understanding is that the transition to Agile will be a significant change for the team, and that there is resistance, particularly from the senior developers. However, by addressing their concerns, providing appropriate training and support, and implementing Agile practices gradually and thoughtfully, this transition can be successfully managed. The goal is to help the team see the benefits of Agile for themselves, so that they will be motivated to embrace this new way of working.

$ Scrum practicalities are difficult to follow. Team does not understand the importance of sprint planning and daily scrums. Team size is five persons plus product owner who is also having scrum master role. As the whole organization moved to use Scrum also this team needs to follow the process. Organization supports scrum masters but not individual team members. Senior developers want buy the new thinking and want to stick in old way of working.

ChatGPT

Understood. It seems like there's a larger organizational push for Scrum and Agile methodology, but the team, particularly the senior developers, are finding it hard to accept this change. The issues of not understanding the importance of sprint planning and daily scrums, coupled with the lack of organizational support for individual team members, are also significant challenges.

Here's a revised action plan considering these issues:

Action Plan

Phase 1: Diagnosis and Communication

1. Clearly communicate the reasons for the move to Scrum from the organizational perspective, explaining the benefits it brings and why it's essential for the entire team to embrace it.

2. Host a session where team members can openly express their reservations and challenges with Scrum. This session could include the product owner/Scrum master and the senior developers. The goal is to listen to their concerns and understand their resistance.

Phase 2: Training and Understanding

1. Arrange for more intensive Scrum training sessions focusing specifically on sprint planning and daily scrums, emphasizing the value and purpose of each. Use practical examples to illustrate their effectiveness.
2. Assign some time to explain Scrum's roles and responsibilities, particularly the product owner and Scrum master's roles. The team needs to understand that the Scrum master is not a traditional "manager" but a "servant leader" who helps the team work effectively.
3. Organize a few workshops or learning sessions about Scrum featuring internal or external speakers who have successfully implemented Scrum. Hearing real-life success stories can motivate the team and help them see the benefits of Scrum.

Phase 3: Gradual Implementation and Mentoring

1. Start with one or two Scrum practices at a time. For example, begin with daily stand-up meetings and sprint planning. Once the team gets comforTable, introduce other elements like sprint reviews and retrospectives.
2. Pair up the senior developers with team members who are more accepting of Scrum, allowing them to observe and experience the benefits firsthand.
3. Provide individual coaching sessions for team members who need additional support in understanding or adopting Scrum.

Phase 4: Feedback and Improvement

1. After each sprint, conduct a thorough retrospective where the team can openly discuss what worked and what didn't. Use this feedback to continuously improve and adapt the Scrum implementation to the team's needs.

2. Recognize and appreciate the team members when they embrace Scrum values and principles. This will encourage others to follow suit.
3. Keep communicating the importance of Scrum and the benefits it brings to the team and the organization.

The key to this process is patience and persistence. Transitioning to Scrum and Agile methodologies requires a shift in mindset which can take time, especially for those accustomed to traditional methods. The goal should be to create a supportive environment where learning and growth are encouraged, and where each team member feels valued and heard.

Model: GPT-4

Maintaining Agile way of working prompt:

$ As a scrum master your task is to maintain agile way of working. The team has the following challenges: Team does not do proper sprint planning, no retrospectives are made, daily scrums do not have value for the team, team is changing the process without any change management. Ask the necessary questions to fully understand the challenges and create coaching plan for maintaining agile way of working. Explain your understanding and provide plan to me representing the project manager. Provide output in notes.

ChatGPT

Understanding the Challenges:

1. **No proper sprint planning:** Without a detailed plan for each sprint, it's impossible for the team to understand their individual responsibilities and the scope of their tasks.
   - **Questions:**
     - How do you currently decide which stories or tasks are included in the sprint?
     - Is there a backlog refinement process in place?
     - Are the user stories clear and have well-defined acceptance criteria?

        - How are team members assigned tasks?
        - How much time is allocated for sprint planning?
        - Do you have a Sprint Goal?

2. **No retrospectives are being made:** Retrospectives are a key part of Agile, providing the team with an opportunity to discuss what went well and what didn't in order to improve future sprints.
    - **Questions:**
        - Have you ever conducted retrospectives in the past?
        - If yes, why were they stopped?
        - If no, are you familiar with the concept and purpose of retrospectives?
        - Do you feel there is a culture of open communication and feedback in the team?
3. **Daily scrums are not providing value:** Stand-up meetings are crucial for keeping everyone on track, addressing blockers, and fostering communication. If they're not adding value, something is going wrong.
    - **Questions:**
        - How is the daily scrum currently structured?
        - What topics are usually discussed?
        - Are all team members actively participating?
        - Is there a time constraint on these meetings?
        - What are the current barriers to effective daily scrums?
4. **Process changes without change management:** In Agile, changes should be managed and communicated effectively to ensure everyone is on the same page and to mitigate potential disruption.
    - **Questions:**
        - Can you give me examples of recent process changes?
        - Why were these changes made?
        - How were these changes communicated to the team?
        - Did you observe any issues or resistance related to these changes?

Coaching Plan:

1. **Improve Sprint Planning:**
    - Conduct training sessions on the importance and benefits of Sprint Planning.
    - Introduce techniques for effective backlog refinement.
    - Set clear Sprint Goals to align the team’s efforts.
    - Ensure user stories have clear, tesTable acceptance criteria.
2. **Implement Retrospectives:**
    - Train the team on the purpose and process of conducting retrospectives.
    - Set up a respectful and safe environment for sharing feedback.
    - Regularly schedule retrospectives at the end of each sprint.
    - Make sure to follow up on action items discussed during retrospectives.
3. **Enhance the Value of Daily Scrums:**
    - Reiterate the purpose and structure of daily scrums (i.e., what was done yesterday, what will be done today, any blockers).
    - Encourage active participation from all team members.
    - Limit the meeting to 15 minutes and focus on the progress towards the Sprint Goal.
    - Address and remove any identified barriers to effective scrums.
4. **Establish Change Management Process:**
    - Define a process for introducing changes to the team's working process, including clear communication, reasons for the change, and the expected impact.
    - Involve the team in the change process to promote buy-in and understanding.
    - Regularly review changes to identify any potential improvements or reversions.

This plan aims to address the identified issues in a structured manner, improving the team's agile working process and overall productivity.

$ Team does not see the point in sprint planning as senior developers think they can do it individually. Each person already has some role and no crossteam roles are not seen important. I as senior developer do not see any big difference working as we have used to.

ChatGPT

Understanding your context and concerns better, it seems like there's a lack of appreciation for the principles of Agile within your team, particularly the benefits of collaboration, cross-functionality, and role flexibility. Below are suggested steps to address these:

Addressing Lack of Value in Sprint Planning

1. **Reiterate the Benefits of Collective Planning:** Conduct a session to highlight the benefits of collective sprint planning. Use data and case studies from other projects to illustrate the benefits such as improved productivity, better quality, fewer missed dependencies, and enhanced team morale.
2. **Showcase the Risk of Siloed Working:** Explain the risks associated with individual planning like the "bus factor" (i.e., what happens if that senior developer gets sick or leaves? What if they make an error in their estimations or overlook a task?). The risk is higher when knowledge and planning are concentrated in a few individuals.
3. **Encourage Collective Ownership:** Agile is not just about completing tasks but about achieving a collective goal. Stress the importance of the whole team owning the sprint goal and how this can't be achieved without collective planning.

Addressing Lack of Cross-team Roles

1. **Promote the Benefits of Cross-functionality:** Showcase how cross-functionality can lead to more flexible, resilient teams, where members can cover for each other and share the load more evenly. Explain that it allows the team to continue working smoothly even when one person is unavailable.
2. **Create Opportunities for Cross-skilling:** Encourage senior developers to mentor other team members and vice versa. This promotes knowledge sharing and cross-skilling, making the team more adapTable.
3. **Celebrate Success of Cross-functionality:** When team members do step out of their usual roles and contribute to the team's success, make sure to acknowledge

and celebrate it. This can help to build a culture where cross-functionality is valued.

Addressing Resistance to Agile Practices

1. **Acknowledge the Concerns:** Recognize the concerns and reservations of the senior developers. They may feel that the Agile practices are unnecessary, time-consuming, or undermining their expertise. Understand where they are coming from and provide reassurances.
2. **Gradual Introduction:** Implement Agile practices gradually. Start small, gather feedback, adapt, and keep iterating. This way, team members have a chance to see the benefits first hand without feeling overwhelmed.
3. **Involve the Team in the Process:** Rather than imposing practices, involve the team in decision-making. Ask for their suggestions on how to improve collaboration, transparency, and efficiency.

It is important to understand that Agile is not just about processes and tools, but also about individuals and interactions. This transformation might take some time and patience, but it is likely to lead to a more productive, resilient, and enjoyable working environment.

Model: GPT-4

Effort estimation prompt:

$ As a senior developer your task is to create an effort estimation. The user story has the following requirements: User needs to login to the system, she needs to enter her email address and password, password needs to be unique and is stored in to nosql db, password needs to be change every 30 days and cannot be reused. Ask the necessary questions to fully understand the requirements and create effort estimation and required tasks. Explain your understanding and provide effort estimation to me representing the scrum master. Provide output in Table containing tasks and effort needed.

ChatGPT

Sure, before giving an estimation, I'd like to clarify a few points:

1. Authentication system: Do we already have an existing authentication system that we can leverage, or do we need to create one from scratch?
2. User interface: Will I be developing the UI for the login, registration and password change functionality, or will it be provided?
3. NoSQL DB: Which specific NoSQL database are we using? MongoDB, DynamoDB, Couchbase?
4. Uniqueness of the password: When we say the password needs to be unique, does it need to be unique for each user or unique globally, i.e., no two users can have the same password?
5. Password change reminder: How should the user be reminded to change their password every 30 days? Via email or any other medium?
6. Password reuse policy: How many previous passwords should the system remember to prevent reuse?

For the purpose of this estimation, let's assume that we are creating the system from scratch, the UI will be provided, we are using MongoDB as the database, the password should be unique for each user, users will be reminded via email, and the system should remember the last five passwords.

Here is the rough task breakdown with estimations:

Task	**Description**	**Effort (in hours)**
Task 1: Designing the DB schema	The schema needs to include user details and a field for storing password history.	4
Task 2: Implementing the registration system	This includes backend logic for creating a new user, validating email, and storing password.	8

Task	**Description**	**Effort (in hours)**
Task 3: Implementing the login system	This includes backend logic for authenticating user credentials.	6
Task 4: Implementing password change system	This will allow users to change passwords, with checks for password history.	8
Task 5: Implementing password change reminder system	This will involve a scheduled process that sends out email reminders.	8
Task 6: Unit Testing	Creating unit tests for each implemented module.	10
Task 7: Integration Testing	Testing the complete flow of registration, login, password change, and reminders.	6
Task 8: Deployment and Documentation	Writing necessary documentation and deploying the system.	4
Total:		**54 hours**

This estimation assumes that the tasks will be completed sequentially. However, in a team setting, some tasks can be worked on in parallel. The above times are rough estimates and might need to be adjusted based on actual implementation and unforeseen complexities.

Keep in mind, these estimates do not include time needed for communication, backlog grooming, meetings, or any blockers that might occur during the development. It's often recommended to add an extra buffer of about 20% for such activities. So, you might want to consider a total of around 65 hours for this user story.

$ We use existing AA in backend which can be conFigured for our purpose. So the topic is actually only designing the web UI. Password has minimum length of 12 characters and should have at least two special characters and two capital letters. Uniqueness is stored in the backend so nobody can use password used once. Email verification is used. All pass-words since the beginning are stored. NoSql will be Redis as it is fast and scalable. We use three Redis nosql nodes. Passwords are hashed using HMAC and salting is also needed.

ChatGPT

Thank you for the additional information. Given these details, it sounds like most of the backend functionality is already in place. We are primarily tasked with designing the front-end interface and integrating it with the backend. Additionally, we will need to handle the password validation requirements on the front end.

Here's an updated task breakdown and effort estimation, taking into account these new details:

Task	Description	Effort (in hours)
Task 1: Design User Interface	Based on provided wireframes or requirements, design the UI for login, registration, and password change.	10
Task 2: Implement User Interface	Implement the designed interfaces using a suiTable web framework.	15
Task 3: Front-end Validation	Implement password validation (length, special characters, capital letters) on the front end.	6
Task 4: Integration with AA Backend	Create the necessary API calls to integrate the frontend with the existing AA backend for user registration, login, and password change functionality.	12

Task	Description	Effort (in hours)
Task 5: Integration with Email Verification System	Integrate the frontend with the email verification system.	6
Task 6: Unit Testing	Creating unit tests for each implemented module.	8
Task 7: Integration Testing	Testing the complete flow of registration, login, password change, and reminders.	8
Task 8: Deployment and Documentation	Writing necessary documentation and deploying the system.	4
Total:		**69 hours**

As before, this estimation assumes that the tasks will be completed sequentially. In a team setting, some tasks can be worked on in parallel. The above times are rough estimates and might need to be adjusted based on actual implementation and unforeseen complexities.

This estimation does not include time needed for communication, backlog grooming, meetings, or any blockers that might occur during the development. It's often recommended to add an extra buffer of about 20% for such activities. So, you might want to consider a total of around 83 hours for this user story.

Model: GPT-4

Technical knowledge analysis prompt:

$ As an engineering manager your task is to estimate technical knowledge in team to support effort estimation. The project has the following technical objectives: Backend is devel-oped using Java and SQL database, system is running on top of RedHat Linux, we need some sort of DevOps, we should need about 3-4 man months to do the project, existing code base exists so only new functionality need to be developed. Ask the necessary questions to fully understand the technical objectives and create plan how team can fulfill the needed technical skills. Explain your understanding and provide plan for me representing the project manager. Provide output in memo format.

ChatGPT

Subject: Technical Knowledge Estimation for Upcoming Project

Dear [Project Manager],

I hope this message finds you well. In response to your request, I have reviewed the project objectives and have identified key areas where we need to assess our team's technical capabilities. Here are the necessary inquiries to fully understand the technical objectives:

1. Java: What is the specific experience level with Java within the team? Are there any specialized libraries or frameworks (Spring Boot, Hibernate, etc.) used in our existing codebase that the team should be familiar with?
2. SQL Database: What is the team's proficiency in SQL? Are there any specific SQL platforms we're using (MySQL, PostgreSQL, Oracle, etc.)? Are there any existing database schemas or specific database management tasks the team should be familiar with?
3. RedHat Linux: How comforTable is the team with Linux system administration? Specifically, what is their experience with RedHat Linux and its server management?
4. DevOps: What DevOps practices and tools (Continuous Integration/Continuous Delivery, Infrastructure as Code, Docker, Kubernetes, Jenkins, etc.) are currently in place? What is the team's proficiency in these areas?
5. Project Scale: Is the 3-4 man-month estimate based on previous projects of similar scale? Does this estimate consider potential challenges or risks that might arise?
6. Existing Codebase: How familiar is the team with the existing codebase? Will there be any refactoring needed to implement the new features?

Here's a plan to ensure our team can fulfill the needed technical skills:

1. Conduct a skills assessment: Have each team member self-assess and provide feedback on their comfort level with the necessary technologies. This will help identify any skill gaps and areas for improvement.
2. Training and Upskilling: Based on the results of the skills assessment, plan necessary training sessions or workshops to address identified skill gaps. This could involve in-house knowledge sharing sessions, external courses, or workshops.
3. Codebase Onboarding: If any team members are unfamiliar with the existing codebase, ensure they're given the necessary support and documentation to get up to speed.
4. Risk Management: Develop a risk management plan considering potential issues that might arise. Regularly review and update the plan throughout the project.
5. Regular Check-ins: Regularly check in with the team to ensure that the learning and upskilling is effective. Monitor project progress and make adjustments as necessary.
6. Utilize a Mentor-Mentee System: Pair less experienced team members with those who have more expertise. This will expedite the learning process and ensure that knowledge transfer is occurring within the team.

With this plan in place, we should be able to adequately prepare our team for the technical demands of this project. Please let me know if there are any additional details or adjustments needed.

Best regards,

[Your Name] Engineering Manager

$ The integration is using CSV, XML and ASCII files between the systems. These are trans-ferred using sftp. Team has two software engineers and they know the basics but not have done this before. Project should be ready in three months. We don't have any training re-sources and team members need to browse internet and ask from Google.

ChatGPT

Subject: Updated Plan for Technical Skill Enhancement for Upcoming Project

Dear [Project Manager],

Thank you for the additional information provided. It appears that our project has some specific challenges that need to be addressed - primarily, integrating CSV, XML, and ASCII files using sftp, and the limited experience our team members have in these areas. However, given the team's basic knowledge and the vast resources available on the internet, I am confident that we can tackle these challenges successfully.

Here is an updated plan to ensure our team acquires and enhances the necessary technical skills:

1. **Self-learning and Research:** Encourage our team members to leverage online resources for learning. There are various repuTable websites, blogs, forums, and video tutorials available for free, where they can gain deeper insights into file integration using CSV, XML, and ASCII, and transferring files using sftp.
2. **Structured Learning Path:** Guide the team members to follow a structured learning path. A suggested path would be to start with a deep understanding of working with CSV, XML, and ASCII files, followed by sftp. Then move on to integration techniques involving these file types.
3. **Community Participation:** Encourage participation in developer communities like StackOverflow, GitHub, etc., where they can pose their doubts and interact with experienced developers. This way, they will not only find solutions to their issues but also get to learn from experts.
4. **Practical Implementation:** Encourage them to work on mini projects or tasks that involve the use of these technologies. This will give them a hands-on experience and help them understand the practical challenges better.
5. **Peer Learning:** Promote a collaborative learning environment. The team members can share their learnings and solutions to the problems they faced during their self-learning journey. This can be done through weekly knowledge-sharing meetings.
6. **Project Timeline:** Given the learning curve and the 3-month project timeline, I recommend starting with simpler tasks related to the project, gradually moving to

complex tasks as our team members get more comforTable with the technologies. This would require a detailed project plan taking into consideration the learning and upskilling required.

7. **Ongoing Support:** As the engineering manager, I will provide ongoing support to our team members, assisting them in overcoming any challenges they encounter in their self-learning journey and project implementation.

While this approach is less conventional and may present certain challenges, I believe that our dedicated team, supported by a wide array of online resources and a supportive learning environment, will be able to complete the project successfully.

Please share your thoughts on this updated plan.

Best regards,

[Your Name] Engineering Manager

APPENDIX B: PROMPT TEST ROUND 2

Model: GPT-4

Requirement creation prompt:

$ As a requirement engineer your task is to create a requirement specification. The requirement specification must fulfill the following constraints: Internet banking system which can handle typical banking transactions to be possible to browse account and make payments. Ask the necessary questions needed to fully understand the constraints. Based on my answers create requirement specification. Explain your understanding and construct your proposal to me representing the project manager. Provide your output in Table format.

ChatGPT

As a requirement engineer, my role is to gather, analyze, document, and validate the needs and expectations of stakeholders in a software project. In this case, we're dealing with an internet banking system. I'll ask a series of questions to further understand the requirements for this project:

1. What types of banking transactions should the system support?
2. What types of payments should the system support? Are there specific types of transfers or payments the system needs to handle?
3. Should the system support multi-currency transactions?
4. How should account browsing be implemented? What specific information about the account should users be able to view?
5. Should the system provide real-time transaction updates?
6. What security measures should be in place to ensure the safe and secure operation of the internet banking system?
7. What platforms should the system be accessible from? (e.g. web, mobile apps, etc.)
8. Does the system need to integrate with other banking systems or third-party services?

9. What level of performance is expected from the system in terms of speed, reliability, and uptime?
10. What type of users should the system support? (e.g. individual customers, businesses, bank administrators)
11. How should the system handle errors and exceptions?
12. Does the system need to support accessibility standards?

Once I receive the answers to these questions, I will be able to create a detailed requirement specification.

Based on the initial constraints given, I'll make an initial requirement specification proposal as follows:

Requirement ID	**Requirement Description**	**Requirement Type**	**Priority**
R1.1	System shall allow users to view their account details	Functional	High
R1.2	System shall display account balance in real-time	Functional	High
R1.3	System shall allow users to make payments	Functional	High
R1.4	System shall support various types of banking transactions (to be defined later)	Functional	High
R2.1	System shall be accessible from web browsers and mobile devices	Non-Functional	High
R2.2	System shall ensure secure transactions by implementing up-to-date security standards	Non-Functional	High
R3.1	System shall provide a user-friendly interface for account browsing and making payments	Non-Functional	High

Requirement ID	Requirement Description	Requirement Type	Priority
R3.2	System shall handle errors gracefully and inform users appropriately	Non-Functional	Medium
R4.1	System shall support integration with other banking systems or third-party services (if required)	Functional	Medium

This Table is an initial proposal based on the constraints given. We'll refine and expand this list based on the answers to the questions above.

$ Users are retail customers and end user customers. Account can be used for checking, savings and credit cards transactions. Payment types should be related to ecommerce so credit cards and money transfer. Recurring payments should be provided. Security follows normal banking security. MFA should be supported through mobile app. The system is integrated with ecommerce shop. GDPR regulations apply. User need to have bank account statements report of the transactions made. Customer support should have chat and normal telephone. In addition FAQ is good to have.

ChatGPT

Based on your responses, I've further refined the requirements. Here are the updated requirements in Table format:

Requirement ID	Requirement Description	Requirement Type	Priority
R1.1	System shall allow retail and end-user customers to view their account details	Functional	High
R1.2	System shall support checking, savings, and credit card transactions	Functional	High

Requirement ID	Requirement Description	Requirement Type	Priority
R1.3	System shall display account balance in real-time	Functional	High
R1.4	System shall allow users to make payments through credit cards and money transfers	Functional	High
R1.5	System shall support recurring payments	Functional	High
R1.6	System shall provide bank account statements detailing transactions made	Functional	High
R2.1	System shall be accessible from web browsers and mobile devices	Non-Functional	High
R2.2	System shall implement banking-level security, including multi-factor authentication (MFA) via a mobile app	Non-Functional	High
R2.3	System shall comply with GDPR regulations	Non-Functional	High
R3.1	System shall provide a user-friendly interface for account browsing, making payments, and viewing account statements	Non-Functional	High
R3.2	System shall handle errors gracefully and inform users appropriately	Non-Functional	Medium
R4.1	System shall support integration with eCommerce shops	Functional	Medium
R4.2	System shall offer a chat and telephone-based customer support system, and provide FAQs for users	Functional	Medium

These requirements cover the basic needs for the internet banking system as described in your response. As we continue refining the project, we can adjust or add to these requirements as needed.

Model: GPT-4

Steering group prompt:

$ As a project's steering group member your task is to fully understand agile principles, business objectives and give feedback about the project. The project has the following objectives: We are implanting an ERP integration system to customer using SAP product. Integration is done in agile way and we expect to release one integration in every two weeks. At the moment we do not get customer to participate demos and would need their feed-back. Project cannot start final testing before customer testing. Customer does not have time for the testing but is expecting the release soon. Ask the necessary questions to fully understand the objectives to be able to understand the project status and way of working. Explain your understanding and provide necessary actions for me as the project manager. Provide output in memo format.

ChatGPT

Subject: Understanding of Project Objectives, Status and Proposed Action Plan

Dear Project Manager,

I am writing to share my understanding of the project and the issues we are facing, as well as to propose several action steps we might consider.

The project involves the implementation of an Enterprise Resource Planning (ERP) integration system for a client using the SAP product. The work is being done in an agile manner with the goal of releasing one integration every two weeks. It appears that we are facing challenges in involving the customer in the demo stages of each sprint, which is essential for collecting valuable feedback and enabling customer testing before the final product launch. Another issue is that the customer is expecting the product to be released soon, but they don't have time to participate in the testing process.

Here are some questions to further understand the context:

1. What communication channels do we currently have with the customer? Is there a dedicated point of contact?
2. Are there any clear reasons why the customer is not participating in the demos? Are they simply unavailable or are there other reasons?
3. Have we communicated the importance and necessity of the customer's feedback and testing to them?
4. What is the customer's understanding of the project timeline? Do they understand the potential impacts on the timeline due to lack of their involvement in testing?
5. Are there any particular features or aspects of the system that the customer is specifically interested in, or concerned about?

Given my current understanding of the project, here are my proposed action steps:

1. **Stakeholder Engagement:** We need to increase the level of customer

$ We have customer representative but she is not so active, however we communicate on weekly basis. We would like to know customer testing that everything is ok and we can proceed to final testing. Customer is expecting the release based on the date given on the contract. With customer we use Teams on weekly basis to share also documentation.

ChatGPT

Subject: Enhanced Stakeholder Engagement and Action Plan for Successful Project Completion

Dear Project Manager,

Thank you for your response and for clarifying the current state of engagement with the customer. It appears that while communication is ongoing, we need to work towards making it more effective to ensure that project objectives are met.

Understanding that the customer representative is not as active as required and the need for customer testing to proceed to the final stages, as well as the customer's expectations for release date as per the contract, here are the revised action steps:

1. **Boosting Engagement:** Consider scheduling a dedicated session with the customer representative to discuss the importance of their active participation in the process. We could clarify how their input and testing directly impact the quality of the final product and adherence to the agreed timelines.
2. **Effective Communication:** Use our weekly Teams meetings to share a clear roadmap of the project and highlight where their inputs are crucial. This may include milestones, testing phases, and the risk of potential delays due to non-participation.
3. **Testing Facilitation:** Propose an easy and time-efficient testing process for the customer. This could involve presenting a specific section of the system to test during each meeting, providing written guidance on what to look for, and offering support to answer any questions that might arise during the testing.
4. **Contract Review:** Examine the existing contract to identify if there is room for renegotiation of timelines based on the current challenges. It is crucial that any potential impact on the delivery date is communicated as soon as possible.
5. **Documentation Sharing:** Continue to share documentation via Teams, making sure they are concise, clear, and address the specific needs of the customer. Use these documents as a basis for discussion in the meetings.

By addressing these steps, we can work towards greater involvement from the customer and ensure that the project moves forward effectively, while managing expectations and maintaining the quality of the final product.

Best Regards,

[Your Name]

Model: GPT-4

Role clarification prompt:

As an engineering manager your task is to specify clear roles for the team. The project has the following constraints: software is created using C++ and publicly known open source components. Only one person knows how to program in C++ but the rest 4 persons have experience only in Python and Javascript. Team needs to deliver C++ application in Linux environment in the next four months. Ask all necessary questions to fully understand the project and be able guide team to create necessary role specifications. Explain your under-standing and provide solution to me representing the management. Provide output in Table format.

ChatGPT

Sure, to better understand the constraints of the project and identify the roles for the team, I need the following information:

1. What is the scale of the project? How large or complex is the C++ application expected to be?
2. What open-source components are you planning to use? Are they C++ libraries or Python/Javascript components?
3. What is the nature of the application? For example, is it a GUI application, a console application, a web application, a system tool, or something else?
4. Does the application need to interface with any other systems or databases?
5. Will there be any need for front-end development? If so, is this a web-based front end or a native GUI?
6. Do the Python and JavaScript developers have experience with any tools or libraries that could facilitate the use of C++ (like Cython or Emscripten)?
7. Do the Python and JavaScript developers have experience in developing applications for a Linux environment?
8. What level of testing is required for the project? Unit tests, integration tests, system tests?
9. Is there any specific software development methodology the team follows?

10. Are there any infrastructure or DevOps tasks required as part of the project, such as setting up development, staging, or production environments, or implementing continuous integration/continuous deployment pipelines?
11. What is the nature of the software's users? Will there be a need for extensive documentation or user training materials?

Once I receive answers to these questions, I will be able to provide a clear role specification for your team members.

Assuming I have all these details, here's a high-level role specification that might fit your situation, subject to refinement:

Role	Responsibility
C++ Developer	Develop core components of the C++ application, and mentor the Python and Javascript developers in C++.
Python/Javascript Developers (2)	Learn C++, assist in building parts of the C++ application, or work on interfacing tasks using libraries like Cython or Emscripten. They could also be tasked with building any web front-end (if applicable).
Python/Javascript Developer (Test)	Learn C++, contribute to development, but mainly focus on building the test suite. They would be responsible for writing unit, integration, and system tests.
Python/Javascript Developer (DevOps)	Set up the development, testing, and production environments, and manage any continuous integration/continuous deployment pipelines. If applicable, they could also handle interfacing with any databases or other systems.

These roles, of course, will be fine-tuned based on the nature of the project and the skills of the team members. It's also worth noting that everyone will need to have some understanding of C++ and the Linux environment, as the project will be developed and delivered in this context. Thus, some form of training or self-study might be necessary for the Python and Javascript developers.

$ Project needs to deliver C++ application in embedded device and transfer the data to backend. It should collect IoT data and move that to the backend for further processing. We use existing cloud based backend but IoT device as Atmel based 32 bit processor and necessary hardware. We would like to utilized existing sw designers also in C++ development. We collaborate through github using its features. Linux is ubuntu based. Testing is done fully manually as we don't have suiTable tools for testing C++ applications and the application is simple. We plan to make some error updates but otherwise maintenance is approx two times in the year.

ChatGPT

Thank you for providing the additional details. Based on this information, I have refined the roles and responsibilities for your team. Given the IoT and embedded system nature of your project, a good understanding of low-level C++ programming is crucial. You mentioned that the team will use an existing cloud-based backend and the Python and JavaScript developers could contribute significantly in this area. It's also essential that team members become familiar with GitHub and understand its features for effective collaboration.

Here's the refined role specification for your team:

Role	**Responsibility**
C++ Developer (Embedded Systems Specialist)	Develop the core components of the C++ application for the IoT device, focusing on low-level programming and hardware interfacing, and mentor the Python and Javascript developers in C++.

Role	Responsibility
Python/Javascript Developers (2) (Backend Specialists)	Assist in learning C++, especially as it pertains to IoT and embedded systems. Develop the interface between the IoT device and the cloud backend, using the skills and experience they have with Python and Javascript.
Python/Javascript Developer (Manual Tester)	Learn C++ to understand the application, and develop and execute a thorough manual test plan for the C++ application. They will be responsible for finding, documenting, and communicating issues to the team.
Python/Javascript Developer (DevOps/Release Engineer)	Manage the version control system (GitHub), coordinate releases, and handle any required post-release maintenance. They could also be tasked with setting up a suiTable Linux (Ubuntu-based) development environment for the team.

This approach has several benefits:

1. **Leveraging existing skills**: The Python/Javascript developers will be using their existing skills as much as possible. For instance, they could help in interfacing the device with the cloud-based backend.
2. **Learning and growth**: The Python/Javascript developers will learn C++, which could be valuable for them in the future.
3. **Shared responsibility**: With the C++ developer acting as a mentor, the C++ learning curve for the Python/Javascript developers will be smoother.

This plan's success will depend on the team's willingness to learn and collaborate, and on effective communication within the team. It's also crucial to give the team enough time and resources to upskill in C++ and embedded systems. Depending on the application complexity, four months might be a tight schedule, and this should be communicated clearly to management. It may be necessary to bring in additional C++ expertise or extend the project timeline depending on how quickly the team can learn and apply their new C++ skills.

Model: GPT-4

Redundancy analysis prompt:

$ As a resource manager your task is to analyze redundancy in agile team skills. The project team has the following persons: one Java back-end programmer, test automation developer, two JavaScript front-end programmers and DevOps specialist. Responsibilities of the team are: create full-stack application for netbanking system, UI is based on javascript and backend is done using Java. Ask the necessary questions to fully understand the task and create analysis of redundancy gaps in the team. Explain your understanding and provide solution to me representing the project manager. Provide output in Table.

ChatGPT

Certainly. In order to get a better understanding of the situation and create a detailed analysis of redundancy gaps in the team, I'll need answers to a few questions:

1. What are the critical tasks or competencies required by the project? Please specify for both the front-end and back-end development, as well as for DevOps and testing.
2. Do the front-end developers know Java, or does the back-end developer have knowledge of JavaScript? Can they work interchangeably to some extent?
3. Does the test automation developer have skills in both Java and JavaScript, in order to test both front-end and back-end effectively?
4. Does the DevOps specialist have sufficient knowledge in both Java and JavaScript environments?
5. Are there any specific frameworks or libraries that the team will be using for this project, such as React, Angular, Vue for JavaScript, and Spring, Hibernate, etc. for Java?
6. What level of understanding does the team have regarding the business domain (netbanking)? Can they back up each other in case of need?

7. Can the team members take over tasks from one another in case of absence due to sickness or holidays?

Once you provide these details, I will be able to create a more precise analysis.

After receiving answers to the above questions, we can categorize the possible redundancy gaps into 4 main areas: Front-end development, Back-end development, Testing, and DevOps.

Here is a general overview, which can be fine-tuned based on your specific answers:

Team Member	**Front-End Development (JavaScript)**	**Back-End Development (Java)**	**Test Automation**	**DevOps**
Java Back-End Programmer		Primary	Secondary	
Test Automation Developer	Secondary	Secondary	Primary	
JavaScript Front-End Programmer (1)	Primary		Secondary	
JavaScript Front-End Programmer (2)	Primary		Secondary	
DevOps Specialist				Primary

- Primary: The team member's main role and responsibility.
- Secondary: The team member has enough skills to support this role if needed.

If you observe the Table, you can notice that there are several gaps in redundancy. Here are some potential solutions:

1. **Cross-training**: Involve team members in cross-training. Front-end developers could learn Java and back-end developer could learn JavaScript to support each other. The DevOps specialist could also be involved in the development process to some extent.
2. **Upskill the test automation developer**: If the test automation developer could upskill to work with DevOps or gain skills in front-end or back-end development, it would be beneficial for the team's redundancy.
3. **Hire an additional member**: Depending on the scale and duration of the project, hiring a full-stack developer could solve the redundancy problem. This developer could work on both front-end and back-end tasks as needed and support any team member in their absence.

Please note that these are generalized solutions, and specific solutions may vary based on the details of the team's skills and project requirements.

$ Team members are doing their own job and have not been helping in tasks having unknown skills. Two senior developers have done in the past versatile programming with different programming languages. Generally the knowledge of programming is quite good. It would be good that persons could do more cross-functional development. Netbank is quite complex as it needs UI and backend knowledge in security and banking domain. Project is ongoing and releases are around four times per year. Only one person can do the backend at the moment. You have good questions and I cannot give answer for each.

ChatGPT

Thank you for providing additional context. It is quite common in agile teams for individuals to specialize in a particular area, but having some degree of cross-functionality certainly increases the team's flexibility and resilience. Given your team's configuration and the additional information you provided, here's a more refined analysis:

Team Member	**Front-End Development (JavaScript)**	**Back-End Development (Java)**	**Test Automation**	**DevOps**	**Security & Banking Domain Knowledge**
Java Back-End Programmer	Secondary	Primary	Secondary		Primary
Test Automation Developer	Secondary	Secondary	Primary		Secondary
JavaScript Front-End Programmer (1)	Primary	Secondary	Secondary		Secondary
JavaScript Front-End Programmer (2)	Primary	Secondary	Secondary		Secondary
DevOps Specialist				Primary	Secondary

- Primary: The team member's main role and responsibility.
- Secondary: The team member has enough skills to support this role if needed.

Your team shows a strong need for cross-functional skill development. Based on the new information you provided, the following steps can be taken to improve the redundancy in skills and to make the team more resilient:

1. **Cross-Training**: Initiate cross-training programs for team members. The two senior developers seem to have potential to extend their skills to other areas based on their past experience. Given the complexity of the netbanking system, it would be helpful if at least one of the front-end developers could also handle back-end tasks when required.

2. **Knowledge Sharing Sessions**: Arrange regular knowledge sharing sessions where each team member shares their work and challenges with the rest of the team. This would help in understanding each other's roles better and also in gaining a better understanding of the overall system.
3. **Domain Training**: Since netbanking is a complex domain, organize specific training on this domain. A better understanding of the banking and security aspects would help the team members in designing and developing better solutions.
4. **Pair Programming**: Encourage pair programming practices where two developers work on the same code. This will not only improve the code quality but also helps in skills transfer among the team members.
5. **Backup Plan**: Identify a secondary person for each role. They can step in to support or take over the primary person's tasks when required. This could be facilitated through cross-training and pair programming.

These measures should help the team to become more flexible and versatile, covering more areas of the project and reducing the dependency on any single team member.

Model: GPT-4

Agile process coaching pattern:

$ As an agile coach your task is to coach agile processes. The project has the following objectives: build a web application to be ready in autumn market. We have backlog full of features. Some of the features are quite abstract. Job needs to be done in two weeks iterations and we should be able to test and release the software as often as possible. Project team is five persons with senior and junior designers not familiar of Scrum or agile ways of working. Ask the necessary questions to fully understand the objectives and create agile coaching plan to the team. Explain your understanding and show me the plan representing the project manager. Provide output in action plan format.

ChatGPT

Sure, I understand the brief you've given and I'll be asking a few questions to gather more information that'll help me provide an effective coaching plan for your team.

1. **Clarification on Objectives**: The primary objective is to build a web application ready for the autumn market, correct? Are there any secondary objectives I should be aware of?
2. **Backlog and Features**: You mentioned that your backlog is full of features, some of which are quite abstract. Are these features ranked by priority? Do we have any details on their dependencies?
3. **Testing and Release Cycle**: You want to be able to test and release the software as often as possible. Do you currently have a Continuous Integration/Continuous Deployment (CI/CD) system in place?
4. **Team Familiarity with Agile**: The team has a mix of senior and junior designers, none of whom are familiar with Scrum or agile practices. Do they have any experience with any other project management methodologies?
5. **Product Owner and Stakeholders**: Who is the Product Owner in this scenario? Who are the other key stakeholders we should engage with?
6. **Tools and Platforms**: What tools do you currently use for project management, version control, and communication? Are you open to recommendations on this?

Based on the information provided, here's an initial action plan to get you started with Agile:

Week 1: Agile Introduction and Training

1. **Agile/Scrum Training**: Conduct an interactive training session to introduce the team to the principles and values of Agile and Scrum. Include an overview of roles, ceremonies, artifacts, and rules.
2. **Scrum Roles**: Assign Scrum roles – Product Owner, Scrum Master, and Development Team.
3. **Tool Setup**: Ensure all team members are familiar with the tools that will be used for project management, version control, and communication.

Week 2: Backlog Refinement and Sprint Planning

4. **Backlog Refinement**: Work with the Product Owner to refine and prioritize the backlog. Break down abstract features into smaller, manageable user stories.

5. **Sprint Planning**: Conduct the first sprint planning meeting to select the work for the first sprint.

Weeks 3-4: Sprint Execution and Agile Coaching

6. **Sprint Execution**: Start the first sprint. Regularly check in with the team to ensure they are comforTable with the new process.
7. **Agile Coaching**: Provide coaching to the team throughout the sprint, addressing any queries and helping resolve any impediments.
8. **Daily Scrum**: Facilitate daily scrum meetings to check progress and remove impediments.

End of Week 4: Sprint Review and Retrospective

9. **Sprint Review**: At the end of the sprint, hold a review meeting to demonstrate the work done and get feedback.
10. **Sprint Retrospective**: Conduct a retrospective to identify what went well, what didn't, and what could be improved in the next sprint.

This is a high-level plan and can be adjusted based on specific circumstances and the team's agility in adopting the Scrum framework. The key is to instill a culture of continuous improvement and flexibility to adapt based on feedback and learning. The iterative and incremental approach will also ensure that the product is built to the highest quality standards, meeting the needs and expectations of stakeholders.

$ All team members are dedicated to their function. Team has tried to use Scrum but with no good luck. As being mostly remotely team has been used Teams with adhoc meetings and some daily startup meetings. This is important project for the company as it is the first web application to the market. Time line for project is the next four months. We would like to release after each sprint but the sprint working is new. So we don't have experience on Scrum practicalities. The only challenge in our team is the level of knowledge.

ChatGPT

Thank you for the additional information. It is good to know that the team has some familiarity with Scrum, albeit with unsuccessful results in the past. The team's commitment and the importance of the project to the company will provide strong motivation for a successful transition to Agile. Below, you'll find a revised action plan which incorporates your feedback:

Week 1: Agile Introduction, Reiteration, and Training

1. **Agile/Scrum Training**: Even though the team has some exposure to Scrum, start with an interactive training session to reintroduce the team to Agile principles and Scrum practices. Focus on practical applications of Scrum in their day-to-day tasks.
2. **Scrum Roles**: Reiterate the roles of Product Owner, Scrum Master, and Development Team. Ensure everyone understands their responsibilities.
3. **Tool Setup**: Confirm everyone is comforTable with Teams for communication and meetings. Introduce a tool for project management and backlog maintenance (like Jira or Trello) and make sure everyone is trained to use it.

Week 2: Backlog Refinement and Sprint Planning

4. **Backlog Refinement**: Collaborate with the Product Owner to refine the backlog. Ensure features are transformed into clearly defined user stories.
5. **Sprint Planning**: Conduct the first sprint planning meeting to decide the work for the first sprint.

Week 3-4: Sprint Execution and Agile Coaching

6. **Sprint Execution**: Begin the first sprint. Provide coaching to the team during the sprint, addressing any questions and helping remove impediments.
7. **Daily Scrum**: Facilitate daily stand-up meetings using Teams. Ensure these meetings are structured, short, and focused.

End of Week 4: Sprint Review, Retrospective, and Release

8. **Sprint Review**: Hold a review meeting to showcase the work done and get feedback.
9. **Sprint Retrospective**: Conduct a retrospective to find out what went well, what didn't, and what could be improved for the next sprint.
10. **Release**: Plan a release after each sprint. Make sure the team understands this is a goal but not a requirement if the increment is not ready.

Week 5 onwards: Repeat and Improve

11. **Sprint Cycle**: Repeat the sprint cycle (planning, execution, review, and retrospective) for each two-week sprint. Make improvements based on the learnings from the retrospectives.
12. **Ongoing Coaching**: Provide ongoing coaching and support. Foster a culture of continuous learning and improvement. Keep reinforcing the principles of Scrum and Agile.

The timeline for this action plan matches your project timeline of four months, with the first week for training, and then two-week sprints. At the end of this period, your team should be much more comforTable with Agile and Scrum, and the project should be ready for its autumn market launch. Please keep in mind that Agile adoption is a journey and it is normal to face challenges along the way. The key is to learn from each sprint and continuously improve.

Model: GPT-4

Adaptability management prompt:

$ As an agile coach your task is to coach team to adapt and motivate in occurring changes. The project has the following objectives: to deliver ERP system integration to new vendor system, vendor is changing their way of working constantly, team has used to work in two weeks sprints, but the customer is working ad hoc and does not respect sprints. Ask the necessary questions to fully understand the objectives and create a plan how to adapt to the changes. Explain your understanding and provide the actions to me representing the project manager. Provide output in Table.

ChatGPT

Understanding the objectives, gathering necessary information and creating an effective plan will require diving deeper into various aspects of the project and its stakeholders. Here are the questions to be asked:

1. What is the scope of the ERP system integration project?
2. How are the vendor changes communicated and how often do they occur?
3. Why is the customer not respecting the sprint cycle?
4. Are there any specific customer requirements we should be aware of?
5. Does the team have experience with ERP system integration projects?
6. How is the current agile methodology implemented within the team?
7. What is the current level of interaction and collaboration between the team and the customer?
8. Are there any blockers or constraints which can affect the project timeline or the implementation of changes?

Based on the answers to these questions, we can develop a plan to guide the team through the changes.

Understanding and Actions	Steps
Understanding: The scope of the ERP system integration will affect the team's ability to adapt to changes. A large or complex project will require more flexibility and adaptive strategies.	**Action:** Provide the team with comprehensive training on the scope of the project and the requirements of the new vendor system. This will help them better anticipate potential changes and understand how they impact the overall project.
Understanding: If vendor changes are constant, it will disrupt the rhythm of the team. They must be prepared for constant shifts in	**Action:** Implement a change management process to better handle vendor changes. This could include frequent communication with the vendor, thorough

Understanding and Actions	Steps
work and manage their expectations accordingly.	documentation of changes, and quick adaptation and response from the team.
Understanding: The customer's ad hoc work style disrupts the team's sprint cycle. It's essential to understand why the customer is not respecting the sprint cycle and find a mutually beneficial solution.	**Action:** Hold a meeting with the customer to discuss the importance of respecting the sprint cycle. Offer alternative ways of communication like scheduled sprint reviews where they can raise changes or concerns. At the same time, consider adapting the team's work schedule to be more in sync with the customer's ad hoc requests, such as introducing a Kanban approach along with Scrum, forming a "Scrumban" methodology.
Understanding: The team's experience and familiarity with the agile methodology will play a big role in their adaptability. If they are well-versed in agile practices, they will be more capable of handling the changes.	**Action:** Reinforce agile principles within the team, ensuring they understand the reasons behind agile practices and the benefits they offer. Provide additional training if necessary.
Understanding: Effective interaction and collaboration with the customer is vital for the success of the project.	**Action:** Facilitate regular communication between the team and the customer. This will not only allow the team to respond quickly to changes, but also build a stronger relationship with the customer.
Understanding: Any existing blockers or constraints could potentially delay the project or hinder the implementation of changes.	**Action:** Regularly assess potential blockers or constraints and devise strategies to overcome them. This could involve improving team collaboration, resolving technical issues, or streamlining processes.

Remember, the key is to foster a culture of agility and continuous improvement within the team. This will enable them to adapt to changes more effectively and maintain high productivity levels, regardless of external changes.

$ Vendor changes happens quite ad hoc but typically in week or two. ERP used is SAP and persons have some knowledge of that as end users. We have weekly meetings over Teams with the vendor. Customer is not aware of vendor schedule as there is not direct channel between them. Customer is expecting releases on time independently on agile system used. We stick in the own release cycle but there is pressure to change that towards customer's schedule. Team is not motivated when lots of unplanned changes are coming.

ChatGPT

Thanks for providing the additional information. Based on your responses, here's an updated understanding and action plan.

Understanding and Actions	**Steps**
Understanding: Vendor changes are communicated with a reasonable frequency but are ad-hoc, making it difficult for the team to anticipate them.	**Action:** Develop a better line of communication with the vendor. Ensure that they are aware of your sprint schedules and ask if it's possible for them to provide a roadmap of changes. Even if the changes are not always known in advance, having a dedicated line of communication can help manage expectations.
Understanding: The team has some familiarity with the SAP system as end users, which can be an advantage but might not be sufficient for integration tasks.	**Action:** Provide additional training or resources to deepen their knowledge of the SAP system and prepare them for integration tasks. This could also include bringing in an expert or consultant if necessary.

Understanding and Actions	Steps
Understanding: The customer is not aware of the vendor's schedule and expects releases to be delivered on time, adding more pressure to the team.	**Action:** Advocate for transparency and open communication among all stakeholders. While it might not be possible to create a direct line of communication between the customer and the vendor, regular updates can be provided to the customer to keep them informed of any vendor changes that might impact the release schedule. Also, negotiate realistic deadlines with the customer based on the frequent changes from the vendor.
Understanding: The team is struggling with motivation due to the constant, unplanned changes.	**Action:** Foster resilience and agility within the team. Reinforce the value of their work, celebrate wins, big or small, and encourage open discussion about challenges. It's also crucial to help the team understand the reasons behind the constant changes and how they can handle them better. Additionally, ensure the team has the support they need to manage their workloads effectively.
Understanding: There is pressure to align the team's release cycle with the customer's schedule.	**Action:** If the team's and customer's schedules are significantly misaligned, it might be beneficial to explore different project management methodologies. For instance, a shift towards a more flow-based approach like Kanban, or a blend of Scrum and Kanban (Scrumban), can allow for more adaptability in managing the incoming changes while still preserving the structure of Agile.

This plan should help your team better manage the incoming changes, align better with the customer's expectations, and maintain their motivation and productivity. Remember, change is a constant in Agile environments, and the team's ability to adapt and thrive amid these changes is crucial.

Model: GPT-4

Change resistance prompt:

$ As an agile coach your task is to improve team way of working to overcome change resistance. The project team has the following challenges: team has been working in water-fall method over five years, scrum or agile way of working is new to the team, senior developers would like to continue the work as they have used to without dailies and retrospectives. Ask the necessary questions to fully understand the challenges and provide plan how to overcome change resistance. Explain your understanding and thinking to me representing the scrum master in the team. Provide output in action plan format.

ChatGPT

Understanding the Challenges:

1. **Status Quo Bias:** The team has been using the waterfall model for over five years. This long-term familiarity with the method means the team could be comforTable with its predictability and routine, resulting in resistance to the change.
2. **Unfamiliarity with Agile:** Since the team has never worked in an Agile or Scrum environment, they might be unsure about the new processes and roles involved, and how it will affect their work.
3. **Resistance from Senior Developers:** Senior developers have expressed a desire to maintain current practices, potentially out of a concern that Agile practices like daily standups and retrospectives could disrupt their workflows or productivity.

Given these challenges, a gradual transition that involves extensive communication, education, and collaborative goal-setting may be most effective. The transition should be tailored to the team's unique needs and concerns to minimize disruption and maximize buy-in.

Action Plan:

1. **Educate and Communicate:**
   - **Action:** Organize workshops to introduce Agile and Scrum methodologies, emphasizing their benefits and explaining how they differ from the

Waterfall method. Highlight successful case studies within the organization or from industry to illustrate these points.

   - **Objective:** To equip the team with the knowledge they need to understand why this change is beneficial and necessary.
2. **Lead by Example:**
   - **Action:** Incorporate Agile practices in the team meetings and communication. As an Agile coach, demonstrate the use of Agile techniques in your interactions.
   - **Objective:** To show the team that Agile methodologies can be beneficial in real-world contexts and aren't as disruptive as they may fear.
3. **Incremental Adoption of Agile Practices:**
   - **Action:** Introduce Agile practices gradually instead of implementing all at once. Start by introducing stand-ups and retrospectives and explain their importance.
   - **Objective:** To provide a smooth transition and avoid overwhelming the team with multiple changes at once.
4. **Address Concerns of Senior Developers:**
   - **Action:** Have one-on-one conversations with the senior developers to understand their concerns better. Explain how Agile practices can potentially benefit their work, and discuss compromises that could make the transition easier for them.
   - **Objective:** To minimize resistance from influential team members and ensure they feel heard and valued.
5. **Promote a Culture of Openness and Collaboration:**
   - **Action:** Foster an environment where team members feel comforTable sharing their concerns, challenges, and suggestions related to the transition. Use retrospectives to address these issues and adapt the approach as necessary.
   - **Objective:** To make the team feel involved in the change process and more willing to embrace the new methodologies.
6. **Feedback and Improvement:**

  - **Action:** Regularly seek feedback about the Agile implementation and make necessary adjustments. Celebrate small wins to motivate the team and show them the progress they're making.
  - **Objective:** To continually refine the transition process, addressing problems quickly, and celebrating success to keep morale high.

By adopting this gradual, understanding-focused approach, the team may become more comforTable with Agile methodologies over time, making a complete transition more feasible.

$ Scrum practicalities are difficult to follow. Team does not understand the importance of sprint planning and daily scrums. Team size is five persons plus product owner who is also having scrum master role. As the whole organization moved to use Scrum also this team needs to follow the process. Organization supports scrum masters but not individual team members. Senior developers want buy the new thinking and want to stick in old way of working.

ChatGPT

Understanding these challenges, it's clear that the team is facing difficulty in understanding the relevance and benefits of Scrum practices. Here are additional steps we can take to help ease this transition:

Action Plan:

1. **Reinforce the Value of Scrum Practices:**
   - **Action:** Run focused workshops to explain the purpose and benefits of each Scrum event (like the Sprint Planning and Daily Scrum), and how they contribute to the team's productivity and product quality. Emphasize that these practices are not merely process requirements, but tools for their success.
   - **Objective:** To ensure the team understands why these practices are important and how they can benefit from them.

2. **Role Clarification:**
    - **Action:** Clarify the roles and responsibilities within a Scrum team, particularly the Product Owner and Scrum Master roles. If the Product Owner is also acting as the Scrum Master, discuss with them about the potential challenges it might bring and how to navigate them. Possibly, consider bringing in a separate Scrum Master to ensure proper focus on both roles.
    - **Objective:** To clear up potential confusion about roles and to ensure that both Product Owner and Scrum Master responsibilities are fully addressed.
3. **Mentoring and Coaching:**
    - **Action:** Pair the team with experienced Scrum mentors or coaches who can guide them in understanding and implementing Scrum practices effectively. This could involve shadowing or pair-programming with experienced Scrum team members from other teams.
    - **Objective:** To provide the team with practical, hands-on guidance in Scrum practices.
4. **Senior Developers Engagement:**
    - **Action:** Involve senior developers in decision-making processes where applicable, and ensure their concerns are being heard and addressed. Illustrate how Scrum practices can enhance software development, project management, and customer satisfaction.
    - **Objective:** To help senior developers see the value in the new practices and encourage their buy-in.
5. **Escalate the Need for Team Support:**
    - **Action:** Advocate for individual team members at an organizational level, emphasizing the importance of supporting all team members during this transition, not just Scrum Masters. This could involve discussions with management or HR about providing additional training or resources to the team.
    - **Objective:** To ensure the organization provides adequate support to all team members, which is essential for a successful transition to Scrum.
6. **Incremental Change:**

  - **Action:** Implement Scrum practices incrementally rather than all at once. Start with the most foundational events like Sprint Planning and Daily Scrum. Once the team gets comforTable, introduce other Scrum events and artifacts.
  - **Objective:** To avoid overwhelming the team and provide them time to adjust to each new practice.

By addressing these specific issues, we can hopefully improve the team's understanding and acceptance of Scrum practices, paving the way for a more successful transition.

Model: GPT-4

Maintaining Agile way of working prompt:

As a scrum master your task is to maintain agile way of working. The team has the following challenges: Team does not do proper sprint planning, no retrospectives are made, daily scrums do not have value for the team, team is changing the process without any change management. Ask the necessary questions to fully understand the challenges and create coaching plan for maintaining agile way of working. Explain your understanding and provide plan to me representing the project manager. Provide output in notes.

ChatGPT

Understanding the Challenges

1. Team does not do proper sprint planning:

- Can you elaborate on what you mean by "proper" sprint planning?
- What issues are you currently encountering with your existing sprint planning process?
- How is the backlog currently managed?
- How are stories and tasks estimated and assigned?
- Is there a clear definition of "Done" that all team members understand and follow?

2. No retrospectives are made:

- Why aren't retrospectives being conducted currently?
- What is the team's understanding and attitude towards retrospectives?
- Are there concerns or obstacles preventing retrospectives from taking place?
- What kind of feedback mechanisms are in place currently?

3. Daily scrums do not have value for the team:

- What's the general feedback from the team about daily scrums?
- What is the current format of your daily scrums?
- Are there challenges with respect to time management, communication or relevance of topics discussed?
- What outcomes does the team expect from these meetings?

4. Team is changing the process without any change management:

- What changes have been made recently, and what prompted them?
- How are these changes being communicated and documented?
- Is there resistance or confusion due to these changes?
- What is the impact of these changes on team performance and project outcomes?

Coaching Plan for Maintaining Agile Way of Working

1. **Sprint Planning:**
    - Conduct a workshop to re-emphasize the importance and benefits of sprint planning.
    - Establish a clear, shared definition of "Done".
    - Improve the backlog management process to ensure all items are clearly defined and prioritized.
    - Implement methods for effective story estimation (like planning poker).
    - Ensure every team member has a clear understanding of their tasks for the upcoming sprint.
2. **Retrospectives:**
    - Explain the value of retrospectives to the team and encourage open and honest communication.
    - Establish a safe, blame-free environment for conducting retrospectives.

    - Introduce various retrospective techniques to keep it engaging and productive.
    - Make sure the team understands that retrospectives are an opportunity to improve, not a forum for blaming or criticizing.
3. **Daily Scrum:**
    - Encourage every team member to come prepared and ensure they speak about what they did yesterday, what they plan to do today, and if there are any blockers.
    - Reinforce the purpose of daily scrums to synchronize activities and create a plan for the next 24 hours.
    - Address issues with time management and keep meetings short and to the point.
    - Modify the format if necessary to meet the team's needs and expectations.
4. **Change Management:**
    - Establish a transparent process for proposing, reviewing, and implementing changes.
    - Encourage team members to communicate any proposed changes and their implications.
    - Incorporate changes into the retrospective discussion or a separate change management meeting.
    - Document all changes and ensure that all team members understand and agree with them.

These steps will help reinforce Agile principles and practices, improve team communication, and make the Agile process more effective and beneficial for the entire team.

$ Team does not see the point in sprint planning as senior developers think they can do it individually. Each person already has some role and no crossteam roles are not seen important. I as senior developer do not see any big difference working as we have used to.

ChatGPT

Thank you for sharing the team's perspective. It seems the team, especially the senior developers, value individual contributions more than collective decision-making and planning, which may hinder the agility of the team. Let me reframe the approach a bit:

Updated Understanding

1. **Team does not do proper sprint planning:**
    - The team, especially senior developers, believe they can handle tasks individually.
    - There is less importance placed on cross-team roles, possibly due to a lack of understanding of their value.
    - As a senior developer yourself, you're comforTable with the current working mode and may not see the need for change.

Updated Coaching Plan

Sprint Planning:

- **Why it's important**: Explain the value of collective sprint planning to the team, focusing on how it can benefit even experienced developers. Discuss how it reduces dependency on individuals, helps manage risk better, ensures smoother handovers in case of absences, and contributes to overall team growth.
- **How to make it appealing**: Make sprint planning more interactive and less bureaucratic. Use planning poker or similar techniques to estimate tasks, which can encourage discussion and collective decision-making.
- **Highlight the benefits of cross-team roles**: Present cases where cross-skilling has helped teams handle unexpected situations or workload spikes. Discuss how it contributes to personal professional development.

Cultural Shift:

- **Influence the shift**: As a senior developer yourself, you play a crucial role in influencing the team's attitudes. Embracing Agile practices and demonstrating their benefits through your work can encourage others to follow suit.

- **Promote a team mindset**: Foster an environment where the team's success is valued over individual achievement. Share success stories that highlight the benefits of teamwork and collaborative problem-solving.

Remember, any significant change takes time, especially when it requires a shift in mindset. As a Scrum Master, it's crucial to lead by example and continually highlight the value of Agile practices.

Model: GPT-4

Effort estimation prompt:

$ As a senior developer your task is to create an effort estimation. The user story has the following requirements: User needs to login to the system, she needs to enter her email address and password, password needs to be unique and is stored in to nosql db, password needs to be change every 30 days and cannot be reused. Ask the necessary questions to fully understand the requirements and create effort estimation and required tasks. Explain your understanding and provide effort estimation to me representing the scrum master. Provide output in Table containing tasks and effort needed.

ChatGPT

Sure, I'm happy to help you break down this task into individual user stories for effort estimation. However, I need some clarification on a few points to ensure I understand the requirements correctly:

1. When you say the password needs to be unique, do you mean it needs to be unique for each user, or unique each time the user changes it (i.e., it can't be the same as a password they've used before)?
2. For the password change requirement, is there a need for an automated reminder or forced password change, or do we simply not allow passwords older than 30 days?

3. Are there any specific security requirements for password storage and transmission?
4. Is there a requirement for email verification when a user registers or changes their email?
5. Is there a forgot password flow that needs to be implemented?

Once we have clarified these points, I can provide a more accurate estimation.

Assuming your answers, here's an example of how I might break down these tasks:

Task	**Description**	**Estimated Effort (in hours)**
Task 1	Design database schema for users in NoSQL database.	4
Task 2	Implement secure password storage using a suiTable hash+salt method.	8
Task 3	Implement user registration system, including password requirements and email input.	12
Task 4	Implement login system, including appropriate error handling for incorrect emails/passwords.	12
Task 5	Implement system to track last password change date and enforce 30-day password change policy.	8
Task 6	Implement system to prevent re-use of previously used passwords.	8
Task 7	Implement email verification on registration and email change.	8

Task	Description	Estimated Effort (in hours)
Task 8	Implement forgot password flow, including secure password reset.	8
Task 9	Write tests to ensure all the above features work correctly.	12
Task 10	Manual testing and bug fixing.	20

This list should cover most of the work involved in implementing this user story. Of course, these are just rough estimates, and the actual effort required could be more or less depending on the specific details of your project and the capabilities of your development team. Remember, it's essential to review and adjust estimates as work progresses.

Keep in mind that there will also be other tasks that need to be done in parallel, such as setting up the necessary development environments, handling DevOps tasks, and planning and attending scrum meetings. These are not included in this list, but they're also an important part of the development process.

$ We use existing AA in backend which can be conFigured for our purpose. So the topic is actually only designing the web UI. Password has minimum length of 12 characters and should have at least two special characters and two capital letters. Uniqueness is stored in the backend so nobody can use password used once. Email verification is used. All pass-words since the beginning are stored. NoSql will be Redis as it is fast and scalable. We use three Redis nosql nodes. Passwords are hashed using HMAC and salting is also needed.

ChatGPT

Great, thanks for providing the additional details. Given that your team is focusing solely on designing the web UI and your authentication/authorization system is already in place, I've revised the tasks and estimates below:

Task	**Description**	**Estimated Effort (in hours)**
Task 1	Design the UI/UX for the login and registration pages.	8
Task 2	Implement the user registration UI, including password requirements validation and email input.	8
Task 3	Implement the login UI, including appropriate error handling for incorrect emails/passwords.	8
Task 4	Implement a UI feature to notify users if their password is due to be changed (based on backend data).	4
Task 5	Implement the change password UI and enforce the password policy (length, special characters, capital letters) at the frontend.	8
Task 6	Implement UI for email verification on registration and email change.	8
Task 7	Implement UI for the "forgot password" flow, including secure password reset.	8
Task 8	Connect the frontend UI to the backend authentication/authorization system through APIs.	8
Task 9	Write tests to ensure all the above features work correctly at the frontend.	12

Task	Description	Estimated Effort (in hours)
Task 10	Manual testing and bug fixing.	20

These are rough estimates and will depend on the specific requirements of your project and the proficiency of your developers. These estimations might need to be adjusted as the project progresses.

Remember, apart from the estimated tasks, there are other additional tasks like setting up the necessary development environment, handling DevOps tasks, planning, and attending scrum meetings. These tasks are not included in this estimation but are equally important for a successful project delivery.

Model: GPT-4

Technical knowledge analysis prompt:

$ As an engineering manager your task is to estimate technical knowledge in team to support effort estimation. The project has the following technical objectives: Backend is developed using Java and SQL database, system is running on top of RedHat Linux, we need some sort of DevOps, we should need about 3-4 man months to do the project, existing code base exists so only new functionality need to be developed. Ask the necessary questions to fully understand the technical objectives and create plan how team can fulfill the needed technical skills. Explain your understanding and provide plan for me representing the project manager. Provide output in memo format.

ChatGPT

Subject: Estimation of Technical Knowledge and Skills Development Plan for Upcoming Project

Dear Project Manager,

I am writing to summarize the understanding of the technical objectives for the upcoming project and to propose a plan for the skill development of the team.

Based on your description, the project requires proficiency in:

1. Java for backend development.
2. SQL for database operations.
3. RedHat Linux as the operating system.
4. DevOps for integration and deployment.

Here are some questions that need to be answered to better understand the technical scope and the current team's expertise:

1. Can you elaborate more on the specific Java frameworks or libraries the team should be familiar with?
2. What specific SQL database will be used? Is it MySQL, PostgreSQL, Oracle, or another?
3. What are the expected DevOps tools that we will be using (e.g., Jenkins, Docker, Kubernetes)?
4. How much experience does the team already have with RedHat Linux?
5. Can you provide more information on the new functionality that will be developed? Will it require knowledge of specific libraries, APIs, or technologies?
6. Is there an expected learning curve for the existing codebase? How complex is it and what technologies does it incorporate?

Once these details are clarified, we can better assess the current team's technical skills and identify the gaps.

Given the above, here is a preliminary plan to fulfill the necessary technical skills:

1. **Technical Assessment**: Conduct an internal assessment of the team's skills and knowledge in the specified technologies (Java, SQL, RedHat Linux, and DevOps).

This will involve one-on-one meetings, group discussions, and possibly small technical tests.

2. **Training**: Based on the results of the assessment, organize training sessions for areas where knowledge gaps are identified. These could be in the form of online courses, internal workshops, or external training, depending on the need.
3. **Mentorship**: Pair less experienced team members with those who are more proficient in the necessary areas. This will help them acquire skills quickly and ensure knowledge transfer.
4. **Documentation**: Ensure that the team has access to the necessary documentation for the technologies used. This includes both official documents (like Java/SQL/DevOps official docs) and project-specific documents (like codebase documentation).
5. **Practice**: Create small tasks or projects related to the upcoming project for team members to work on. This will help them apply their new skills and get comforTable with the technologies.
6. **Retrospectives**: Regularly conduct retrospectives to gauge progress, address challenges, and adapt the plan as necessary.

I look forward to your response to the aforementioned questions so we can further refine this plan. Thank you for your attention to this matter.

Best,

[Your Name] Engineering Manager

$ The integration is using CSV, XML and ASCII files between the systems. These are transferred using sftp. Team has two software engineers and they know the basics but not have done this before. Project should be ready in three months. We don't have any training re-sources and team members need to browse internet and ask from Google.

ChatGPT

Subject: Revised Technical Knowledge Estimation and Skill Development Plan

Dear Project Manager,

Thank you for providing the additional details. The project involves integration via CSV, XML, and ASCII files using sftp, and we have a timeline of three months. Considering the team comprises two software engineers with basic knowledge, here's an updated plan to ensure they acquire the necessary skills for this project:

1. **Self-Study**: Given the lack of formal training resources, the team will need to use self-study methods to learn the necessary technologies. This could include reading online tutorials, documentation, and Q&A forums (e.g., StackOverflow), as well as watching video tutorials on platforms like YouTube. The team should focus on the following areas:
   - **Java and SQL**: Deepen their understanding of Java for backend development and SQL for database operations.
   - **Data Formats and Transfer**: Learn about handling and manipulating CSV, XML, and ASCII files in Java, as well as secure file transfers using sftp.
   - **RedHat Linux**: Gain proficiency in RedHat Linux as the operating system, focusing on necessary commands and scripts.
   - **DevOps**: Learn about continuous integration, deployment, and relevant DevOps tools.
2. **Collaborative Learning**: Encourage team members to share knowledge with each other regularly. This could be through presentations on new topics learned, code reviews, or collaborative problem-solving.
3. **Practice Projects**: Create small practice tasks related to the project for the team members. This will give them the opportunity to apply what they've learned and familiarize themselves with the technologies and methodologies involved.
4. **Community Support**: Encourage the team to engage with online technical communities for help. Forums like StackOverflow and GitHub can be invaluable for getting answers to technical questions and learning from others' experiences.
5. **Management Support**: As a manager, you can support the team by allocating adequate time for learning within the project timeline, ensuring the workload is balanced, and resolving any issues that may hinder their learning process.

Please understand that the learning curve may be steep, given the lack of formal training resources and the complex nature of the project. Therefore, regular check-ins and adjustments will be necessary to ensure the team is progressing effectively. This approach may also extend the original timeline.

I hope this plan meets your expectations. Please feel free to provide further feedback or clarification.

Best,

[Your Name] Engineering Manager

APPENDIX C: PROMPT TEST ROUND 3

Model: GPT-4

Requirements creation prompt:

$ As a requirement engineer your task is to create a requirement specification. The requirement specification must fulfill the following constraints: Internet banking system which can handle typical banking transactions to be possible to browse account and make payments. Ask the necessary questions needed to fully understand the constraints. Based on my answers create requirement specification. Explain your understanding and construct your proposal to me representing the project manager. Provide your output in Table format.

ChatGPT

As an AI, I understand that you're looking for a detailed requirement specification for an internet banking system. Before I construct the requirement specification, I'll ask a few questions to ensure I fully comprehend the scope and intricacies of the project:

1. What types of accounts should the system support? For example, checking, savings, credit, loan accounts, etc.
2. What type of payments should the system support? For example, direct transfers, bill payments, mobile payments, international transfers, etc.
3. Should the system offer a view of transaction history? If so, how detailed should this view be?
4. How should users authenticate themselves? Through a username and password, two-factor authentication, biometric authentication, etc.?
5. What level of permissions should different users have? For example, can certain users only view accounts, while others can make transactions?
6. Are there any specific security requirements or regulations the system must adhere to?
7. Does the system need to integrate with any existing systems or third-party services?

8. Should the system support multiple currencies?
9. What's the expected maximum number of users who could be using the system at any given time?
10. Are there any special accessibility requirements?
11. Should the system provide support for mobile devices?
12. What kind of reporting capabilities should the system have?

Based on your answers, I will create the requirement specification for your internet banking system.

$ Users are retail customers and end user customers. Account can be used for checking, savings and credit cards transactions. Payment types should be related to ecommerce so credit cards and money transfer. Recurring payments should be provided. Security follows normal banking security. MFA should be supported through mobile app. The system is integrated with ecommerce shop. GDPR regulations apply. User need to have bank account statements report of the transactions made. Customer support should have chat and normal telephone. In addition FAQ is good to have.

ChatGPT

Based on your answers, I have created the following requirement specification for the internet banking system.

Requirement ID	Description	Category
R1	The system shall support retail and end user customers.	User
R2	The system shall support account types including checking, savings, and credit card transactions.	Functional
R3	The system shall support ecommerce-related payments, including credit card transactions and money transfers.	Functional

Requirement ID	Description	Category
R4	The system shall support recurring payments.	Functional
R5	The system shall adhere to typical banking security standards.	Non-Functional
R6	The system shall support multi-factor authentication through a mobile app.	Security
R7	The system shall integrate with an ecommerce shop.	Integration
R8	The system shall be compliant with GDPR regulations.	Compliance
R9	The system shall generate bank account statements reporting the transactions made.	Functional
R10	The system shall offer customer support via chat and telephone.	Support
R11	The system shall provide an FAQ section for common issues and queries.	Support

It's important to note that these requirements represent a high-level view of what the internet banking system should provide. Each requirement can be broken down into more detailed sub-requirements. As the project moves forward, it will be essential to further refine these requirements and develop a detailed system design.

This specification focuses on the key features of the system. Additional work will be needed in defining other aspects of the system such as the UI/UX, performance requirements, reliability and availability requirements, etc.

Should you have any other requirements or specifications that need to be addressed, please let me know and I'll be glad to assist.

$ I want to add a new requirement where multifactor authentication is based on combination of finger print and username+password. Can you update the requirements?

ChatGPT

Of course, I can add that requirement to the list. Here is the updated requirement specification:

Requirement ID	**Description**	**Category**
R1	The system shall support retail and end user customers.	User
R2	The system shall support account types including checking, savings, and credit card transactions.	Functional
R3	The system shall support ecommerce-related payments, including credit card transactions and money transfers.	Functional
R4	The system shall support recurring payments.	Functional
R5	The system shall adhere to typical banking security standards.	Non-Functional
R6	The system shall support multi-factor authentication through a combination of username/password and fingerprint, using a mobile app.	Security
R7	The system shall integrate with an ecommerce shop.	Integration

Requirement ID	Description	Category
R8	The system shall be compliant with GDPR regulations.	Compliance
R9	The system shall generate bank account statements reporting the transactions made.	Functional
R10	The system shall offer customer support via chat and telephone.	Support
R11	The system shall provide an FAQ section for common issues and queries.	Support

This new requirement (R6) reflects that the system will use multi-factor authentication based on a combination of fingerprint and username/password through a mobile app. This will enhance the security of the system by ensuring users' identities are thoroughly verified before granting them access. As with the previous requirements, this requirement may be broken down into more detailed sub-requirements as the project progresses.

$ FAQ can be removed from the requirements. Instead I need feedback form. Please update and prioritize requirements.

ChatGPT

Certainly, I will remove the FAQ requirement and replace it with a requirement for a feedback form. The prioritization of requirements can be subjective, and it often depends on the business needs, project timeline, and resource availability. However, I'll give you an example of how one might prioritize these requirements, assuming user security and core functionality are the highest priority:

Priority	**Requirement ID**	**Description**	**Category**
High	R1	The system shall support retail and end user customers.	User
High	R2	The system shall support account types including checking, savings, and credit card transactions.	Functional
High	R3	The system shall support ecommerce-related payments, including credit card transactions and money transfers.	Functional
High	R6	The system shall support multi-factor authentication through a combination of username/password and fingerprint, using a mobile app.	Security
High	R8	The system shall be compliant with GDPR regulations.	Compliance
Medium	R4	The system shall support recurring payments.	Functional
Medium	R7	The system shall integrate with an ecommerce shop.	Integration
Medium	R9	The system shall generate bank account statements reporting the transactions made.	Functional
Medium	R12	The system shall provide a feedback form for user input and suggestions.	User
Low	R5	The system shall adhere to typical banking security standards.	Non-Functional

Priority	**Requirement ID**	**Description**	**Category**
Low	R10	The system shall offer customer support via chat and telephone.	Support

Please note that all requirements are important and need to be fulfilled. The prioritization only indicates the sequence or emphasis of implementation, and should be refined in close collaboration with stakeholders.

$ Can you reorder the requirements based on the requirement id?

ChatGPT

Absolutely, here are the requirements sorted by their Requirement ID:

Priority	**Requirement ID**	**Description**	**Category**
High	R1	The system shall support retail and end user customers.	User
High	R2	The system shall support account types including checking, savings, and credit card transactions.	Functional
High	R3	The system shall support ecommerce-related payments, including credit card transactions and money transfers.	Functional
Medium	R4	The system shall support recurring payments.	Functional
Low	R5	The system shall adhere to typical banking security standards.	Non-Functional

Priority	Requirement ID	Description	Category
High	R6	The system shall support multi-factor authentication through a combination of username/password and fingerprint, using a mobile app.	Security
Medium	R7	The system shall integrate with an ecommerce shop.	Integration
High	R8	The system shall be compliant with GDPR regulations.	Compliance
Medium	R9	The system shall generate bank account statements reporting the transactions made.	Functional
Low	R10	The system shall offer customer support via chat and telephone.	Support
Medium	R12	The system shall provide a feedback form for user input and suggestions.	User

Please note that the Requirement IDs are now in order, but the priority of the requirements remains the same as in the previous Table.

$ Change priorities R3 becomes medium, R5 becomes High, R12 becomes low.

ChatGPT

Absolutely, I have revised the priorities of the requirements as per your instructions. Here's the updated Table:

Priority	**Requirement ID**	**Description**	**Category**
High	R1	The system shall support retail and end user customers.	User
High	R2	The system shall support account types including checking, savings, and credit card transactions.	Functional
Medium	R3	The system shall support ecommerce-related payments, including credit card transactions and money transfers.	Functional
Medium	R4	The system shall support recurring payments.	Functional
High	R5	The system shall adhere to typical banking security standards.	Non-Functional
High	R6	The system shall support multi-factor authentication through a combination of username/password and fingerprint, using a mobile app.	Security
Medium	R7	The system shall integrate with an ecommerce shop.	Integration
High	R8	The system shall be compliant with GDPR regulations.	Compliance
Medium	R9	The system shall generate bank account statements reporting the transactions made.	Functional
Low	R10	The system shall offer customer support via chat and telephone.	Support

Priority	Requirement ID	Description	Category
Low	R12	The system shall provide a feedback form for user input and suggestions.	User

Please let me know if there are any other changes you'd like to make.

$ are you able to add more details to each requirement?

ChatGPT

Sure, I can add a bit more detail to each requirement. Please note that these are still relatively high-level requirements; each one can be further broken down into more granular sub-requirements as needed.

Priority	Requirement ID	Description	Category	Details
High	R1	The system shall support retail and end user customers.	User	The system will have different user roles and permissions for retail and end users, with an interface tailored to each type of user.
High	R2	The system shall support account types including checking, savings, and credit card transactions.	Functional	The system will allow users to view balances, transfer funds, and perform transactions related to checking, savings, and credit card accounts.

Priority	Requirement ID	Description	Category	Details
Medium	R3	The system shall support ecommerce-related payments, including credit card transactions and money transfers.	Functional	The system will have the functionality to integrate with ecommerce platforms to allow seamless payments through credit card transactions and direct money transfers.
Medium	R4	The system shall support recurring payments.	Functional	The system will allow users to set up and manage recurring payments for bills or any other regular transactions.
High	R5	The system shall adhere to typical banking security standards.	Non-Functional	The system will follow industry-standard security practices like encryption, secure communication protocols, secure data storage, and more.
High	R6	The system shall support multi-factor authentication through a combination of username/password and fingerprint, using a mobile app.	Security	The system will require users to verify their identity using both their username/password and their fingerprint when using the mobile app.
Medium	R7	The system shall integrate with an ecommerce shop.	Integration	The system will connect with the specified ecommerce platform to facilitate payments and transfer of funds.

Priority	**Requirement ID**	**Description**	**Category**	**Details**
High	R8	The system shall be compliant with GDPR regulations.	Compliance	The system will follow all relevant GDPR guidelines, particularly around data storage, processing, and user consent.
Medium	R9	The system shall generate bank account statements reporting the transactions made.	Functional	The system will allow users to view, download, and print statements for any given period, showing all transactions made.
Low	R10	The system shall offer customer support via chat and telephone.	Support	The system will include an integrated chat function for instant support and a contact number for telephone support.
Low	R12	The system shall provide a feedback form for user input and suggestions.	User	The system will include a simple, easy-to-use feedback form for users to submit comments, concerns, or suggestions.

This should provide a deeper level of understanding for each requirement, however please remember that these can be further detailed and analyzed as the project evolves.

Model: GPT-4

Steering group prompt:

$ As a project’s steering group member your task is to fully understand agile principles, business objectives and give feedback about the project. The project has the following objectives: We are implanting an ERP integration system to customer using SAP product. Integration is done in agile way and we expect to release one integration in every two weeks. At the moment we do not get customer to participate demos and would need their feedback. Project cannot start final testing before customer testing. Customer does not have time for the testing but is expecting the release soon. Ask the necessary questions to fully understand the objectives to be able to understand the project status and way of working. Explain your understanding and provide necessary actions for me as the project manager. Provide output in memo format.

ChatGPT

Subject: Understanding and Recommendations for ERP Integration Project

Dear [Project Manager's Name],

As a steering group member, I have reviewed the current status and the objectives of the ERP integration project. I understand that we are working to integrate an ERP system for a customer using a SAP product, following agile methodologies with the aim of releasing one integration every two weeks. I also understand that the final testing of the project cannot commence until the customer completes their testing phase.

Upon considering these factors, I have a few clarifying questions that will help me fully comprehend the project's scope, status and challenges:

1. **Project timeline and deadlines:** What is the current project timeline and the expected deadlines for the integration? How aligned are we with these timelines?
2. **Client's involvement:** What specific feedback are we seeking from the customer during the demos? Are they familiar with the agile process and the importance of their participation in it?
3. **Customer's testing:** What does the customer testing entail? Is it acceptance testing or functional testing? Who are the key people from the customer's side responsible for this?

4. **Team structure and capacity:** Can you provide some details on our project team? How are tasks and roles divided within the team?
5. **Communication channel with customer:** How have we been communicating with the customer so far? What's their feedback on the process and the project?

Based on the details provided, here are my initial observations and recommendations:

Observations:

- The project seems to be facing a challenge in getting the necessary feedback and participation from the customer. This could potentially delay the start of our final testing and subsequent integrations.
- There might be a gap in communication or understanding between us and the customer about the agile process, and their role in it.

Recommendations:

1. **Education & Advocacy:** We may need to help the customer understand the value and necessity of their involvement in demos and testing phases. This could include an explanation of the agile process, their role, and the impact on the overall project if they do not participate as needed.
2. **Flexible Participation:** If time is a constraint for the customer, we could explore other ways to collect their feedback, such as recording demos for them to view at their convenience, or arranging for shorter, more frequent check-ins.
3. **Define Clear Roles & Responsibilities:** If not done already, it would be beneficial to clearly define and communicate roles and responsibilities, not just for our internal team, but also for the customer. This could help establish expectations and encourage active participation from the customer.
4. **Risk Management:** Based on the customer's current engagement level, we might need to revisit the project risk management plan. It would be good to prepare for scenarios where the customer's input may not be as timely as desired.

I am confident that with open and frequent communication, and by addressing the customer's concerns and constraints, we can ensure the successful completion of the project within the stipulated time frame. I look forward to your response on my questions and feedback on the recommendations.

Best regards,

[Your Name] [Your Position]

$ We have customer representative but she is not so active, however we communicate on weekly basis. We would like to know customer testing that everything is ok and we can proceed to final testing. Customer is expecting the release based on the date given on the contract. With customer we use Teams on weekly basis to share also documentation.

ChatGPT

Subject: Updated Recommendations for ERP Integration Project

Dear [Project Manager's Name],

Thank you for providing additional context. I now understand that we have a designated customer representative who isn't very active, and we're in weekly communication with her via Teams. The purpose of customer testing is to confirm that everything is working as expected so we can proceed to our final testing. I also understand that the release date is contractually agreed upon.

Based on these updates, here are my revised observations and recommendations:

Observations:

1. The engagement level of the customer representative seems to be a significant obstacle to obtaining necessary feedback and advancing to the final testing phase.
2. The contractual obligation for a specific release date increases pressure on the project, making customer feedback and testing even more critical to stay on schedule.

Recommendations:

1. **Improved Engagement:** Identify ways to engage the customer representative more actively in the project. This could involve ensuring that the representative understands the significance of her role, the testing process, and the impact of potential delays on the project timeline. This could also involve providing incentives for active participation or exploring ways to accommodate her schedule or work style.
2. **Alternative Feedback Channels:** If time or availability is a challenge for the customer representative, consider providing alternative means for her to give feedback. For example, we could use asynchronous communication methods like email or a shared document where she can leave comments at her convenience.
3. **Identify Additional Representatives:** If possible, ask the customer if they could assign additional representatives to the project. This can help distribute the workload and provide a backup plan in case the primary representative is unavailable.
4. **Early and Continuous Testing:** To mitigate the risk of the customer discovering issues at a late stage, implement an early and continuous testing strategy. This would involve testing smaller pieces of the system as they are developed and sharing these test results regularly with the customer.
5. **Proactive Risk Management:** Given the constraint of a fixed release date, we should prepare contingency plans for potential delays or setbacks. This could involve identifying portions of the project that can be postponed if needed, or arranging for additional resources to expedite the final testing phase if the customer approval comes in later than expected.

I am confident that with these adjustments, we can enhance the customer engagement, ensure timely feedback, and meet the contractually agreed release date.

Best regards,

[Your Name] [Your Position]

$ Related to team, we have three persons where one person is tester, one person project manager and one software designer.

ChatGPT

Subject: Further Recommendations for the ERP Integration Project

Dear [Project Manager's Name],

Thank you for sharing the details about the project team. Understanding that our team comprises a project manager, a tester, and a software designer allows for more precise recommendations.

Observations:

Given the team's size and composition, it seems crucial to ensure that roles and responsibilities are explicitly defined and followed to maintain efficient collaboration and progress. Additionally, given the customer's current level of involvement, the project team may face challenges in getting timely feedback for testing and moving towards the final testing phase.

Recommendations:

1. **Maximizing the Roles:**
    - **Project Manager:** Given the customer's representative's lack of active participation, the project manager might need to take a more proactive role in managing this relationship. This could include finding more effective ways of engaging the representative, negotiating the involvement of additional customer representatives, and ensuring that customer testing and feedback are received timely.
    - **Tester:** The tester should continue to ensure early and continuous testing to identify and fix issues promptly. Regular communication with the customer representative can also be helpful in making sure the customer is aware of our testing progress and can provide feedback when possible.

   - **Software Designer:** The designer might need to be more involved in the communication with the customer representative, especially when explaining technical aspects of the integration and addressing any concerns or issues the representative might have.
2. **Customer Engagement:** Given the size and roles of our team, it would be beneficial if all team members participate in engaging the customer representative. This would distribute the customer engagement workload and ensure that the representative receives timely and comprehensive support and communication from us.
3. **Leverage Technology:** Utilize project management and collaboration tools to maintain transparency and efficient collaboration among the team members and with the customer representative. These tools can also help track the project's progress and the customer's feedback.
4. **Risk Management:** In light of the team's size, it's essential to have a risk management plan in place that addresses potential issues, such as a team member becoming unavailable. This plan should outline contingency measures to ensure that the project stays on track.

These recommendations aim to facilitate more efficient collaboration within our team and with the customer representative, ultimately leading to a successful and timely project completion.

Best regards,

[Your Name] [Your Position]

$ one correction: we have also support from our customer engagement manager

ChatGPT

Subject: Adjusted Recommendations for the ERP Integration Project

Dear [Project Manager's Name],

Thank you for the additional information regarding the involvement of a customer engagement manager in the project. This information certainly provides more context and opens up more avenues for effective communication and collaboration.

Observations:

Having a customer engagement manager in the project indicates an additional resource and a key role that can be harnessed to ensure better customer participation. This person could serve as a bridge between the customer and our project team, facilitating communication and ensuring that expectations are met on both ends.

Recommendations:

1. **Customer Engagement Manager's Role:** The customer engagement manager should work closely with the customer representative, ensuring they understand their role and importance within the agile process. This person can also gather feedback, arrange meetings or demos, and discuss any challenges or concerns the representative might have. Essentially, the customer engagement manager can serve as an advocate for the project within the customer's organization.
2. **Team Collaboration:** Encourage all team members, including the customer engagement manager, to collaborate closely. They should share updates, challenges, and achievements regularly. The project manager, tester, software designer, and customer engagement manager each have unique insights that, when combined, can lead to a more successful project.
3. **Risk Management:** Given the additional resource of the customer engagement manager, reassess the risk management plan. This person's involvement could potentially help mitigate some of the previously identified risks, such as low customer engagement or delayed feedback.
4. **Project Timeline Monitoring:** The customer engagement manager should actively monitor the project timeline in relation to the customer's expectations and the contractually agreed release date. Any potential delays or issues should be communicated to the project team as early as possible to allow for adjustments and to keep the project on track.

5. **Regular Synchronization:** Schedule regular synchronization meetings with the project team and the customer engagement manager to ensure everyone is on the same page regarding project status, risks, and the next steps.

With the inclusion of the customer engagement manager and the collaborative efforts of the team, I am confident that we can enhance customer engagement and keep the project aligned with the agreed timeline.

Best regards,

[Your Name] [Your Position]

Model: GPT-4

Role clarification prompt:

$ As an engineering manager your task is to specify clear roles for the team. The project has the following constraints: software is created using C++ and publicly known open source components. Only one person knows how to program in C++ but the rest 4 persons have experience only in Python and Javascript. Team needs to deliver C++ application in Linux environment in the next four months. Ask all necessary questions to fully understand the project and be able guide team to create necessary role specifications. Explain your under-standing and provide solution to me representing the management. Provide output in Table format.

ChatGPT

Before we can start defining clear roles for the team, I need to understand the project a bit better. Here are some questions I'd like to clarify:

1. What is the nature of the C++ application? (e.g., web application, desktop application, library, etc.)
2. Does the project involve building a completely new application from scratch, or enhancing an existing one?

3. What are the specific technical requirements for the application? (e.g., performance, security, memory management, etc.)
4. How complex is the application? Are there any existing project documentation or specifications we can reference?
5. Do we need to consider the integration with other systems or APIs?
6. What are the key functionalities or features that need to be implemented?
7. Is there any specific open-source component that the team needs to use?
8. How is the work going to be managed and tracked? Is there a specific methodology (e.g., Agile, Waterfall) you'd prefer to use?
9. Is there any available training budget/time for the Python and JavaScript developers to learn C++ or relevant skills?
10. Is there a possibility to hire a temporary C++ developer to fill the skill gap if necessary?
11. Are there any specific industry or regulatory standards the application needs to comply with?

Based on the initial information you provided and without the specific details above, here's a tentative role specification Table. Please note that the roles may be adjusted depending on the responses to the above questions.

Role	**Tasks**	**Team Member**
Project Manager	Oversees project progress, liaises with stakeholders, manages project risks and issues, coordinates with the team.	To be determined
Senior C++ Developer	Leads the development of the C++ application, provides technical guidance to the team, conducts code reviews, creates technical documentation.	C++ Programmer

Role	Tasks	Team Member
Python/Javascript Developer (Trainee C++ Developer)	Assists the C++ Developer, learns C++ to contribute to the project, continues development using Python/Javascript (for testing, scripting, and auxiliary tools if applicable).	Python/Javascript Programmer 1,2,3,4
Quality Assurance (QA) Engineer	Develops test plans, conducts testing, documents and tracks bugs and issues, ensures that the product meets quality standards.	Can be assigned to one of the Python/Javascript developers if they have QA experience
DevOps Engineer	Manages the Linux environment, conFigures CI/CD pipelines, ensures the software can be reliably built and deployed.	Can be assigned to one of the Python/Javascript developers if they have DevOps experience

Please note that cross-training the Python/Javascript developers on C++ will be critical for this project. This could potentially be a risk, given the 4-month timeline. The Senior C++ Developer will also likely need to spend time mentoring these developers, which could slow the initial development progress. We might want to consider temporary external help, like a contractor, if the timeline or project complexity doesn't allow for sufficient upskilling time.

$ Project needs to deliver C++ application in embedded device and transfer the data to backend. It should collect IoT data and move that to the backend for further processing. We use existing cloud based backend but IoT device as Atmel based 32 bit processor and necessary hardware. We would like to utilized existing sw designers also in C++ development. We collaborate through github using its features. Linux is ubuntu based. Testing is done fully manually as we don't have suiTable tools for testing C++ applications and the application is simple. We plan to make some error updates but otherwise maintenance is approx two times in the year.

ChatGPT

Thank you for the additional information. The application's nature as an embedded C++ application collecting IoT data and sending it to a cloud backend helps clarify the roles required for the project. Here is the revised role specification Table:

Role	Tasks	Team Member
Project Manager	Oversees project progress, liaises with stakeholders, manages project risks and issues, coordinates with the team.	To be determined
Senior C++ Developer	Leads the development of the C++ application on the Atmel 32-bit processor, provides technical guidance to the team, conducts code reviews, creates technical documentation.	C++ Programmer
Embedded Systems Developer (Trainee)	Assists the C++ Developer in programming the embedded application, learns C++ and embedded systems programming to contribute to the project.	Python/Javascript Programmer 1,2
Backend Developer	Ensures successful integration of the data collected by the IoT device with the existing cloud-based backend.	Python/Javascript Programmer 3
Quality Assurance (QA) Engineer	Develops manual test plans, conducts testing, documents and tracks bugs and issues, ensures that the product meets quality standards.	Python/Javascript Programmer 4

Role	Tasks	Team Member
DevOps Engineer	Manages the Linux (Ubuntu) environment, conFigures CI/CD pipelines in GitHub, ensures the software can be reliably built and deployed.	Can be assigned to one of the Python/Javascript developers if they have DevOps experience

Again, the C++ developer would need to mentor the Python/Javascript developers who are transitioning to C++ and embedded systems development. Given the project's constraints, it might be beneficial to hire a contractor with C++ and embedded systems experience if the timeline becomes too tight.

As for testing, given that it's done manually, having a clear and well-documented test plan is essential, especially since the application is running on an embedded system. Automation could save significant time and reduce the chance of errors, so it may be worth considering the investment in automation tools and skills. But this would depend on the project's constraints and the complexity of the application.

Maintenance being planned only twice a year implies that the system should be robust and thoroughly tested before deployment. Any major bugs found after deployment could disrupt the IoT data collection and affect the backend processing.

Finally, the backend developer's role will be crucial in ensuring that the data collected from the IoT device is successfully integrated into the existing cloud-based backend. They may need to work closely with the C++ developer to ensure that the data is correctly formatted and sent from the device. This developer should have experience with cloud-based systems, IoT data, and potentially networking, to ensure successful data transmission.

$ one clarification. There is not training budget and persons need to learn in doing and by coaching each others.

ChatGPT

Understood, thank you for the clarification. In that case, upskilling the team will be a critical part of this project. The Python/Javascript developers will need to learn C++ and embedded systems programming on-the-job, with the guidance of the Senior C++ Developer. Similarly, the DevOps Engineer role will need to be fulfilled by one of the Python/Javascript developers, who would have to learn how to manage a Linux (Ubuntu) environment and how to conFigure CI/CD pipelines in GitHub.

Please note that this approach may extend the time needed to deliver the project, as learning new skills while doing the project work will likely slow the initial progress. Also, this approach could potentially increase the risk of errors or bugs, given the team's initial unfamiliarity with C++, embedded systems, and DevOps practices.

Here is the updated role specification:

Role	**Tasks**	**Team Member**
Project Manager	Oversees project progress, liaises with stakeholders, manages project risks and issues, coordinates with the team.	To be determined
Senior C++ Developer/Coach	Leads the development of the C++ application on the Atmel 32-bit processor, provides technical guidance and coaching to the team, conducts code reviews, creates technical documentation.	C++ Programmer
Embedded Systems Developer (Trainee)	Assists the C++ Developer in programming the embedded application, learns C++ and embedded systems programming through hands-on work and coaching from the Senior Developer.	Python/Javascript Programmer 1,2

Role	Tasks	Team Member
Backend Developer/Coach	Ensures successful integration of the data collected by the IoT device with the existing cloud-based backend, coaches other team members on backend development as needed.	Python/Javascript Programmer 3
Quality Assurance (QA) Engineer (Trainee)	Develops manual test plans, conducts testing, documents and tracks bugs and issues, ensures that the product meets quality standards, learns QA techniques through hands-on work and coaching from other team members.	Python/Javascript Programmer 4
DevOps Engineer (Trainee)	Manages the Linux (Ubuntu) environment, conFigures CI/CD pipelines in GitHub, ensures the software can be reliably built and deployed, learns DevOps techniques through hands-on work and coaching from other team members.	Can be assigned to one of the Python/Javascript developers if they have some DevOps experience

Please note that the successful execution of this project will require a lot of collaboration and peer-to-peer learning. The team should be encouraged to share knowledge, support one another's learning, and communicate effectively to mitigate the challenges of on-the-job learning.

$ Can project manager do scrum mastering?

ChatGPT

Yes, the Project Manager can also take on the role of a Scrum Master, especially in smaller teams or projects. However, it's important to note that the two roles have distinct responsibilities, and combining them might place a considerable workload on the individual.

The Scrum Master's primary role is to facilitate Scrum practices within the team, remove impediments that the team might face, and ensure effective collaboration. The Scrum Master acts more as a servant-leader, fostering an environment where the team can self-organize and work efficiently.

On the other hand, a Project Manager often has a broader role, managing resources, coordinating with stakeholders, overseeing project risks and issues, and generally being accounTable for the successful delivery of the project.

Given that your team is small and it sounds like the project has a fairly narrow focus, it could work for the Project Manager to also act as Scrum Master. However, it's crucial to monitor workload and prevent burnout.

Here is the updated role specification:

Role	**Tasks**	**Team Member**
Project Manager/Scrum Master	Oversees project progress, liaises with stakeholders, manages project risks and issues, coordinates with the team, facilitates Scrum practices, and removes impediments.	To be determined
Senior C++ Developer/Coach	Leads the development of the C++ application on the Atmel 32-bit processor, provides technical guidance and coaching to the team, conducts code reviews, creates technical documentation.	C++ Programmer
Embedded Systems Developer (Trainee)	Assists the C++ Developer in programming the embedded application, learns C++ and embedded systems programming through hands-	Python/Javascript Programmer 1,2

Role	Tasks	Team Member
	on work and coaching from the Senior Developer.	
Backend Developer/Coach	Ensures successful integration of the data collected by the IoT device with the existing cloud-based backend, coaches other team members on backend development as needed.	Python/Javascript Programmer 3
Quality Assurance (QA) Engineer (Trainee)	Develops manual test plans, conducts testing, documents and tracks bugs and issues, ensures that the product meets quality standards, learns QA techniques through hands-on work and coaching from other team members.	Python/Javascript Programmer 4
DevOps Engineer (Trainee)	Manages the Linux (Ubuntu) environment, conFigures CI/CD pipelines in GitHub, ensures the software can be reliably built and deployed, learns DevOps techniques through hands-on work and coaching from other team members.	Can be assigned to one of the Python/Javascript developers if they have some DevOps experience

Model: GPT-4

Redundancy analysis pattern:

As a resource manager your task is to analyze redundancy in agile team skills. The project team has the following persons: one Java back-end programmer, test automation developer, two JavaScript front-end programmers and DevOps specialist. Responsibilities of the team are: create full-stack application for netbanking system, UI is based on javascript and backend is done using Java. Ask the necessary questions to fully understand the task and create analysis of redundancy gaps in the team. Explain your understanding and provide solution to me representing the project manager. Provide output in Table.

ChatGPT

Certainly. To perform a thorough analysis, I need to clarify a few details. Please answer the following questions:

1. How proficient are each of the team members with the technologies being used in the project? Specifically, is there any overlap in their skills or are their abilities strictly limited to their roles?
2. What are the specific responsibilities of the DevOps specialist in the team? Does the role also include elements of system administration, database administration, and/or security management?
3. What's the extent of the Test Automation Developer's responsibilities? Are they solely focused on writing test scripts, or do they also contribute to software development?
4. How critical is the timeline for this project? Do we need to ensure that we have redundancy for all roles to minimize the impact of any potential unavailability?
5. Is there any existing redundancy in the team? For instance, can the JavaScript programmers work with Java if needed, or vice versa?

Once we have this information, we will be better positioned to analyze the current level of skill redundancy in the team and provide solutions for any gaps.

$ Team members are doing their own job and have not been helping in tasks having unknown skills. Two senior developers have done in the past versatile programming with different programming languages. Generally the knowledge of programming is quite good. It would be good that persons could do more cross-functional development. Netbank is quite complex as it needs UI and backend knowledge in security and banking domain. Project is ongoing and releases are around four times per year. Only one person can do the backend at the moment. You have good questions and I cannot give answer for each.

ChatGPT

Thank you for the additional details. I understand that each team member is primarily focused on their own roles, with some degree of versatility in the two senior developers. The project is ongoing with regular updates and requires knowledge in the UI, backend, security, and banking domains.

Based on this information, I'll prepare an analysis of potential redundancy gaps and propose some solutions.

Here's a simplified Table to outline potential areas of risk and suggested improvements for redundancy:

Role	**Redundancy Risk**	**Potential Solution**
Java Backend Programmer	High	Train one of the JavaScript developers or the Test Automation Developer in Java, especially for the basic tasks. Rely on the senior developers for more complex issues.
JavaScript Frontend Programmers	Low	You have two JavaScript developers, which provides some redundancy. However, consider cross-training the Java Backend Programmer to increase depth.
Test Automation Developer	Moderate	Expand the test automation role to involve more development tasks. Cross-train the DevOps specialist or a developer in test automation.
DevOps Specialist	High	Cross-train at least one developer (preferably a senior one) in the basics of DevOps. This could be managing deployment pipelines or understanding infrastructure management.

It's also advisable to promote a culture of knowledge sharing in your team. Encourage pair programming, code reviews, and regular training sessions. This would allow team members to gradually acquire a wider range of skills, making your team more robust and versatile.

If specific training is needed, consider arranging training sessions with experts or take advantage of online learning platforms.

Remember, redundancy is not about having each team member capable of doing all tasks at the same level. Rather, it's about having a backup, someone who can keep things moving in the short term while the primary resource is unavailable.

$ How would you suggest to backup testing person as there is automatic and manual testing required?

ChatGPT

Backup for a testing role, especially when it involves both automated and manual testing, can be approached from multiple directions:

1. **Cross-training Developers:** Encourage developers to write unit tests and understand the testing frameworks used in your project. This not only helps with redundancy but also improves code quality as developers who write tests often write more tesTable (and generally better) code.
2. **Training Sessions:** Organize regular training sessions where the Test Automation Developer shares knowledge and best practices about both manual and automated testing techniques. These sessions can help other team members understand the essentials of testing.
3. **Pair Testing:** Similar to pair programming, pair testing can be an effective method for knowledge sharing. A developer can pair up with the Test Automation Developer to write and execute test cases. This will increase their familiarity with the testing process and tools.
4. **Role Rotation:** Temporarily rotating roles can be a good way to ensure redundancy. A developer can take over the testing tasks for a sprint or two. This will help them get hands-on experience with testing while giving the Test Automation Developer an opportunity to familiarize themselves with other roles.

5. **Hiring a Part-Time Tester or a Testing Intern:** If your budget al.lows, consider hiring a part-time tester or an intern who can assist the Test Automation Developer and also serve as a backup when needed.

In your redundancy analysis Table, the entry for the Test Automation Developer could look like this:

Role	Redundancy Risk	Potential Solution
Test Automation Developer	High	Cross-train developers in unit and integration testing. Organize regular knowledge-sharing sessions and encourage pair testing. Consider role rotation. If budget al.lows, hire a part-time tester or an intern.

Remember that the goal is to ensure continuity of the project even if the primary tester is unavailable. Each team member does not need to become an expert in testing, but they should be capable of performing the basic testing tasks when necessary.

$ Please provide the final Table with the roles

ChatGPT

Sure, here's the complete redundancy analysis Table for your project team:

Role	Redundancy Risk	Potential Solution
Java Backend Programmer	High	Cross-train one of the JavaScript developers or the Test Automation Developer in Java for basic tasks. Rely on senior developers for complex issues.

Role	Redundancy Risk	Potential Solution
JavaScript Frontend Programmers	Low	Cross-train the Java Backend Programmer to increase depth. Consider providing additional JavaScript training to the Test Automation Developer or DevOps specialist.
Test Automation Developer	High	Cross-train developers in unit and integration testing. Organize regular knowledge-sharing sessions and encourage pair testing. Consider role rotation. If budget al.lows, hire a part-time tester or an intern.
DevOps Specialist	High	Cross-train at least one developer (preferably a senior one) in the basics of DevOps. This could be managing deployment pipelines or understanding infrastructure management.

These recommendations should provide you with a roadmap to enhance redundancy in your team and ensure continuity of work even if any team member is unavailable for a period. It is essential to foster a culture of knowledge sharing and continual learning in your team to accomplish this. Regular training sessions, pair programming or testing, and role rotation can help achieve this goal.

Model: GPT-4

Agile coach pattern:

$ As an agile coach your task is to coach agile processes. The project has the following objectives: build a web application to be ready in autumn market. We have backlog full of features. Some of the features are quite abstract. Job needs to be done in two weeks iterations and we should be able to test and release the software as often as possible. Project team is five persons with senior and junior designers not familiar of Scrum or agile ways of working. Ask the necessary questions to fully understand the objectives and create agile coaching plan to the team. Explain your understanding and show me the plan representing the project manager. Provide output in action plan format.

ChatGPT

Sure, here are some key questions to fully understand the objectives and constraints of the project:

1. What is the expected launch date for the web application?
2. What is the relative priority of the features listed in the backlog?
3. What is the specific role and skill level of each team member?
4. Are there any specific requirements or constraints regarding the technology stack or development environment?
5. How frequently will the team be able to release software updates?
6. What kind of support or infrastructure is available for testing and deployment of the software?
7. What are the means of communication with stakeholders and how often should this occur?
8. What kind of software development tools does the team currently use?
9. How familiar are the team members with the principles of Agile and Scrum?
10. What are the anticipated challenges or risks that could impede the project?

Based on the answers to these questions, an Agile Coaching Plan for this project might look like the following:

Objective: Enable a team of five members, composed of senior and junior designers unfamiliar with Agile or Scrum, to work in two-week iterations to build a web application by the Autumn market launch.

Action Plan:

1. **Training:** Conduct initial training sessions to familiarize all team members with Agile principles and the Scrum framework. This includes explaining the roles in Scrum, the purpose of the events, the artifacts, and the rules.

2. **Backlog Grooming:** Work closely with the team to better understand and break down the features in the backlog into smaller, more concrete user stories. Prioritize these stories based on their value to the project's goals.
3. **Sprint Planning:** Guide the team through the process of selecting user stories for the first sprint based on priority and estimated effort. The team decides how much work they can commit to complete during the upcoming sprint.
4. **Daily Scrum:** Implement daily stand-up meetings to track progress, identify blockers, and keep everyone aligned on tasks and responsibilities.
5. **Iteration and Testing:** Encourage the team to work in iterations, creating potentially shippable increments of software. Support the setup of a continuous integration environment to enable frequent testing and early detection of issues.
6. **Sprint Review and Retrospective:** At the end of each sprint, organize a review to demonstrate the accomplished work to stakeholders. Facilitate a retrospective where the team can discuss what went well, what didn't, and how they can improve in the next sprint.
7. **Continuous Improvement:** Regularly reassess and adjust the team's practices to align with the Agile mindset of continual learning and adaptation.
8. **Communication:** Set up regular communication channels with stakeholders and ensure transparency about the project's progress and any potential risks or impediments.
9. **Risk Management:** Identify potential risks and challenges early on and develop mitigation strategies.
10. **Feedback Loop:** Regularly solicit feedback from the team about the Agile process and make necessary adjustments.

Remember that Agile is about flexibility, collaboration, and continuous improvement, so this plan may need to be adapted as the project progresses. Regularly revisit and revise the plan to ensure it continues to serve the needs of the team and the project.

$ All team members are dedicated to their function. Team has tried to use Scrum but with no good luck. As being mostly remotely team has been used Teams with adhoc meetings and some daily startup meetings. This is important project for the company as it is the first web application to the market. Time line for project is the next four months. We would like to release after each sprint but the sprint working is new. So we don't have experience on Scrum practicalities. The only challenge in our team is the level of knowledge.

ChatGPT

Thank you for providing more information. Here's an updated coaching plan based on your responses:

Objective: Train and guide a dedicated remote team of five members, who have some basic familiarity with Scrum but no practical experience, to work efficiently in two-week iterations to successfully build a web application in four months for the Autumn market launch.

Action Plan:

1. **Remote Working Training**: As the team is mostly remote, conduct a session on best practices for remote working, emphasizing communication, collaboration tools, and time management.
2. **Refresher Scrum Training**: Since the team has tried Scrum before without success, provide a refresher course on Scrum. This will include understanding the Scrum roles, events, artifacts, and values. Stress the importance of each event and make sure each role is understood and accepted by team members.
3. **Backlog Refinement**: Assist the team in breaking down the product backlog items into manageable and understandable user stories, defining clear acceptance criteria for each. Facilitate the team's effort to prioritize these stories based on their value to the project.
4. **Sprint Planning Guidance**: Guide the team on how to properly conduct Sprint Planning. This includes understanding the capacity of the team, selecting user stories for the Sprint based on priority and estimated effort, and creating a definition of "Done" for the Sprint.
5. **Daily Scrum Consistency**: Make sure the team sticks to a consistent schedule for the daily scrum, where progress is shared, and blockers are discussed. As the team is remote, ensure that everyone is comforTable using Microsoft Teams for these daily meetings.

6. **Iterative Development and Continuous Integration**: Advocate for the practice of breaking work into small chunks that can be completed within one Sprint, tested, and integrated into the existing code base. This approach helps to mitigate risks associated with integration.
7. **Sprint Review and Retrospective Facilitation**: At the end of each sprint, guide the team in conducting a Sprint Review and Retrospective. Make sure that these are not neglected, as they are crucial for continuous improvement and adaptation.
8. **Support the Release Process**: Since the team is aiming to release after each Sprint, provide guidance on creating a Definition of Done that includes all steps necessary for a release. This ensures that each increment is potentially shippable.
9. **Communication and Transparency**: Encourage regular communication between team members and stakeholders. Provide a platform for the team to share progress, challenges, and learning.
10. **Coaching on Demand**: Be available to provide individual coaching to team members who need additional support or have specific questions about Scrum practices.
11. **Continuous Learning and Adaptation**: Promote a culture of continuous learning and adaptation, encouraging the team to regularly reassess their practices and make changes as necessary for improvement.

The key to this plan is fostering self-organization within the team and creating an environment where each team member feels empowered to contribute and continuously improve. As the team gains experience with Scrum, they should become more comforTable and effective in using it to manage their work.

$ As team is working mostly remotely and they seem to forget on-site working. What can be done in action plan to improve onsite working?

ChatGPT

Certainly, if there are opportunities for on-site work or hybrid working (a mix of remote and on-site), this can often benefit team communication, bonding, and collaborative problem-solving. Here are some updates to the action plan to encourage and improve on-site working:

1. **Promote Regular On-site Days**: Depending on the team's flexibility, promote certain days of the week where all members can be on-site if possible. Important events such as Sprint Planning, Sprint Review, and Sprint Retrospective could be scheduled on these days for face-to-face interaction.
2. **On-site Workshops**: Plan occasional on-site workshops or training sessions. These could cover Agile principles, Scrum practices, team building activities, etc., and serve as a good opportunity for the team to connect in person.
3. **Office Environment Improvement**: If possible, make improvements to the on-site office environment that make it more conducive to Agile practices. This could include setting up dedicated areas for stand-up meetings, having visible boards for tracking work, etc.
4. **Hybrid Meetings**: For meetings where some members are remote and others are on-site, make sure the necessary technology (like video conferencing tools) is available and functional to include everyone in the conversation.
5. **Social Bonding Activities**: Plan occasional social activities on-site to build team morale and foster stronger relationships. This can be anything from team lunches to recreational activities.
6. **Travel Reimbursement**: If commuting to the office presents a financial burden to team members, consider implementing a travel reimbursement policy to cover these costs.
7. **Health and Safety**: Ensure health and safety measures are in place and communicated to the team. This could be particularly important depending on the ongoing pandemic situation.

Remember that not all team members might be comforTable or able to work on-site due to personal circumstances or health concerns. It's essential to maintain a flexible approach that respects individual needs while also promoting the benefits of occasional on-site collaboration.

Model: GPT-4

Agile process coaching prompt:

$ As an agile coach your task is to coach agile processes. The project has the following objectives: build a web application to be ready in autumn market. We have backlog full of features. Some of the features are quite abstract. Job needs to be done in two weeks iterations and we should be able to test and release the software as often as possible. Project team is five persons with senior and junior designers not familiar of Scrum or agile ways of working. Ask the necessary questions to fully understand the objectives and create agile coaching plan to the team. Explain your understanding and show me the plan representing the project manager. Provide output in action plan format.

ChatGPT

Understood. Before I provide an Agile coaching plan, I would like to clarify a few points:

1. Do the team members have any prior experience with any Agile methodology or any other project management methodology?
2. How complex are the features in the backlog? Could you provide an example of an "abstract" feature?
3. What is the expected duration of the project, i.e., when is the "autumn market"?
4. What are the roles of the team members? Are there dedicated developers, QA, and a product owner?
5. What resources and tools do we currently have to facilitate Agile practices (e.g., issue-tracking systems, CI/CD pipelines, testing frameworks, etc.)?
6. Is there an existing requirement gathering and documentation process in place?
7. How is the communication with stakeholders handled currently?

Based on the given scenario and assuming answers to some of these questions, here is a proposed Agile Coaching plan:

Objective: Implement Agile (Scrum) practices to deliver a web application project for the autumn market release, with bi-weekly iterations and frequent software releases. Increase team's familiarity with Agile methodologies.

Action Plan:

Week 1:

1. **Agile Training:** Conduct a workshop to familiarize the team with the Agile principles and Scrum framework. Explain the roles (Product Owner, Scrum Master, Development Team), artifacts (Product Backlog, Sprint Backlog, Increment), and ceremonies (Sprint Planning, Daily Scrum, Sprint Review, Sprint Retrospective).
2. **Role Assignment:** Assign Scrum roles among team members.
3. **Tool Setup:** Set up necessary tools for project management (like Jira), version control (like Git), CI/CD pipelines (like Jenkins), and communication (like Slack or Teams).

Week 2:

4. **Backlog Grooming:** Work with the team to refine and prioritize the product backlog. Break down abstract features into smaller, actionable user stories. Define acceptance criteria.
5. **Estimation:** Conduct a planning poker session to estimate the effort for each user story.

Sprint 1 and onward:

6. **Sprint Planning:** At the start of each sprint, hold a planning meeting to decide the scope of the sprint based on the team's velocity and product backlog priorities.
7. **Daily Scrum:** Facilitate daily stand-ups to discuss progress and blockers.
8. **Testing & Integration:** Promote test-driven development. Encourage frequent commits and continuous integration of code.
9. **Sprint Review & Retrospective:** At the end of each sprint, facilitate a sprint review meeting to demo the developed features to stakeholders. Conduct a retrospective to reflect on the sprint and identify areas of improvement.

Throughout the project:

10. **Stakeholder Communication:** Maintain active communication with stakeholders. Keep them informed about the project progress and seek their feedback regularly.

11. **Coaching & Mentoring:** Continually coach team members on Agile practices. Promote a culture of collaboration, feedback, and continuous improvement.
12. **Adaptation:** Regularly inspect and adapt the process to better fit the team and project requirements.

At the end of the project:

13. **Project Review:** Conduct a final project review and retrospective to learn from this project and improve future projects.

This is a broad plan and would be tailored to the team's and project's specific requirements once those are clarified. Please let me know if you need more specific actions or any changes to this plan.

$ All team members are dedicated to their function. Team has tried to use Scrum but with no good luck. As being mostly remotely team has been used Teams with adhoc meetings and some daily startup meetings. This is important project for the company as it is the first web application to the market. Time line for project is the next four months. We would like to release after each sprint but the sprint working is new. So we don't have experience on Scrum practicalities. The only challenge in our team is the level of knowledge.

ChatGPT

Thanks for providing the additional information. Given this context, here is the revised Agile coaching plan to implement Scrum:

Objective: Guide the team in successful implementation of Agile (Scrum) methodologies for the development of their first web application, aiming to have bi-weekly release cycles over a four-month timeline.

Action Plan:

Week 1:

1. **Agile Refresher:** Conduct an Agile and Scrum refresher workshop to reinforce understanding and emphasize on the benefits of Scrum. Address questions or misconceptions the team may have from their previous experience.
2. **Roles Clarification:** Review the Scrum roles and their responsibilities, ensuring everyone understands their role and expectations clearly.
3. **Tool Optimization:** Ensure the team is using Microsoft Teams effectively for communication, meetings, and maintaining transparency. Integrate Teams with project management software like Azure DevOps or Jira, and establish a version control system like Git.

Week 2:

4. **Backlog Refinement:** Guide the team in refining and prioritizing the backlog, ensuring that items are clearly defined, actionable, and include defined acceptance criteria. Encourage the team to turn abstract features into user stories.
5. **Estimation Workshop:** Conduct a workshop to train the team on story point estimation and planning poker. This will help the team to estimate the effort for each user story more accurately.

Sprint 1 and onward:

6. **Sprint Planning:** At the start of each sprint, hold a planning meeting. Encourage the team to only commit to the amount of work they feel they can complete based on their velocity.
7. **Daily Scrum:** Coach the team to conduct effective daily stand-ups where each member discusses what they did the previous day, what they will do today, and any blockers they have.
8. **Continuous Integration and Delivery:** Coach the team on effective use of CI/CD tools for regular integration and delivery of code. This will help them prepare for regular releases after each sprint.
9. **Sprint Review & Sprint Retrospective:** At the end of each sprint, facilitate the sprint review and retrospective meetings. Help the team understand how these ceremonies can help them showcase their work and continuously improve their process.

Throughout the project:

10. **Stakeholder Engagement:** Teach the team effective techniques for stakeholder communication and involvement. This will be critical in ensuring stakeholder buy-in and managing expectations.
11. **Coaching & Mentoring:** Provide ongoing coaching and mentoring to the team members on Scrum and Agile practices. This will ensure continuous learning and improvement.
12. **Inspect & Adapt:** Regularly review and improve the Agile practices based on the team's feedback and the project's requirements.

At the end of the project:

13. **Project Review:** Conduct a final project review and retrospective to capture learning from this project. This will help in improving the processes for future projects.

This action plan is focused on reinforcing Scrum principles and enhancing the practical understanding of Scrum ceremonies, roles, and techniques among the team members. It also emphasizes on continuous learning, improvement, and communication, which are key to the success of any Agile project.

$ We have very limited time in project for any additional training. Do you have suggestions how to sneak in these hours in the project?

ChatGPT

Certainly, if time for dedicated training sessions is limited, we can take a more integrated approach. The key idea here is 'learning by doing', where the team gains knowledge about Agile practices while they are actually implementing them.

Here's the adjusted action plan:

Week 1:

1. **On-the-Job Agile Overview:** During the kickoff meeting, spend some time to quickly recap Agile principles and Scrum framework in an engaging manner. Focus on the 'why' behind Agile and Scrum, as understanding the reasons can lead to better implementation.
2. **Roles Clarification:** Discuss Scrum roles and responsibilities during the initial team meetings, ensuring everyone is clear about their role.
3. **Tool Optimization:** Utilize existing team meetings to discuss the effective use of tools. You could provide quick tips and tricks during stand-ups or dedicate a few minutes in the beginning or end of sprint planning/review sessions.

Week 2 onwards:

4. **Backlog Refinement:** Refinement should be an ongoing process. Initially, use the first part of sprint planning sessions to work with the team on backlog refinement. Over time, this could evolve into dedicated refinement meetings as the team gets accustomed to Scrum.
5. **Estimation Learning:** Learn estimation techniques on the job during sprint planning meetings. Start with a simple approach like t-shirt sizes (S, M, L, XL) to indicate effort and complexity, then gradually introduce more sophisticated techniques like planning poker as the team becomes comforTable.

Sprint 1 and onward:

6. **Sprint Planning:** Use the first few sprint planning meetings as hands-on training sessions. Guide the team to self-organize and decide what they can commit based on their capacity.
7. **Daily Scrum:** Coach the team during the initial few stand-ups about the purpose and how to keep the discussion focused and efficient.
8. **Continuous Integration and Delivery:** Introduce and coach on CI/CD practices during the development phase, ideally when the team is actually pushing code and can see the benefits first-hand.
9. **Sprint Review & Sprint Retrospective:** Utilize the first sprint review and retrospective sessions to guide the team about the purpose, structure, and potential outcomes from these meetings.

Throughout the project:

10. **Coaching & Mentoring:** Learning will happen continuously as the team works. Encourage questions and make sure to clarify them promptly. Share helpful resources like articles, blogs, videos that team members can go through at their own pace.
11. **Inspect & Adapt:** Use retrospective meetings to identify what's working and what's not in your Agile implementation. Adapt the process accordingly.

This action plan leverages the team's regular activities for learning, thus making the best use of limited time. As an Agile coach, remember to be patient, encourage open communication, and foster a safe environment where team members feel comforTable asking questions and making mistakes. The team’s Agile maturity will increase with time and practice.

Model: GPT-4

Adaptability coaching prompt:

$ As an agile coach your task is to coach team to adapt and motivate in occurring changes. The project has the following objectives: to deliver ERP system integration to new vendor system, vendor is changing their way of working constantly, team has used to work in two weeks sprints, but the customer is working ad hoc and does not respect sprints. Ask the necessary questions to fully understand the objectives and create a plan how to adapt to the changes. Explain your understanding and provide the actions to me representing the project manager. Provide output in Table.

ChatGPT

First, let's clarify the objectives and the current situation with some questions:

1. What is the scope of the ERP system integration? Are we dealing with full integration, partial integration, or a specific module integration?
2. Can you specify the nature of the changes in the vendor's way of working? Are these changes related to technology, workflows, or business rules?

3. How is the communication managed between the vendor and our team? Do we have any established protocols or channels?
4. How do ad-hoc requests from the customer affect the current sprint work? Can you provide an example?
5. Is there any flexibility in the customer's process to accommodate our sprint-based workflow, or is it strictly ad-hoc?
6. How often do ad-hoc changes come up? Are there any patterns?
7. What is the impact on team productivity and morale due to the current approach?
8. Does the team have any experience with other agile methodologies which might be better suited to handle an ad-hoc work environment?

Based on the answers to these questions, a plan can be created to navigate through these changes. Here is a possible plan:

Understanding	**Action**
The scope of integration is critical. Depending on the level of integration, we need to align our sprints and tasks accordingly.	Clarify the scope with the stakeholders. Plan sprints in a way that accommodates both integration tasks and potential changes.
The vendor's changing methods could lead to miscommunication or confusion.	Establish regular communication with the vendor to understand the changes. Share this information with the team in a timely manner to keep them updated.
The customer's ad-hoc style is disturbing the rhythm of the two-week sprints.	Educate the customer on the benefits of the sprint-based workflow. Negotiate a hybrid model that includes some predictability in requests without eliminating all ad-hoc requests.

Understanding	Action
Frequent interruptions in the current sprint work affect team productivity and morale.	Introduce a buffer or an "unplanned" user story in each sprint to accommodate ad-hoc requests, maintaining productivity without overburdening the team.
The team is not comforTable with the ad-hoc changes from the customer.	Train the team on handling ad-hoc requests efficiently within the sprint structure. Encourage a growth mindset to help them adapt to changes.
If the frequency and pattern of ad-hoc changes can be understood, we can better manage the sprints.	Monitor the frequency and type of ad-hoc changes. Use this information to predict and plan future sprints.
If there is flexibility in the customer's process, we can better align our methodologies.	Open dialogue with the customer to understand their flexibility. Propose changes that can help align both parties' processes.
If the team is familiar with other agile methodologies, we can utilize those to handle the ad-hoc work.	Discuss with the team and identify if other methodologies, such as Kanban, could be utilized in conjunction with Scrum to manage the ad-hoc requests better.

Keep in mind this is just a sample plan. The actual plan might vary based on the answers to the questions and the specific dynamics of your team, vendor, and customer.

$ Vendor changes happens quite ad hoc but typically in week or two. ERP used is SAP and persons have some knowledge of that as end users. We have weekly meetings over Teams with the vendor. Customer is not aware of vendor schedule as there is not direct channel between them. Customer is expecting releases on time independently on agile system used. We stick in the own release cycle but there is pressure to change that towards customer's schedule. Team is not motivated when lots of unplanned changes are coming.

ChatGPT

Based on the new information, we can refine our plan. Let's break down this understanding into actions:

Understanding	Action
The vendor's ad-hoc changes typically occur every one to two weeks, which aligns with the team's current sprint schedule.	Communicate these changes promptly to the team and allocate time within the sprint to address them. Use the weekly Teams meetings to discuss these changes with the vendor in advance, if possible.
The team has some knowledge of SAP as end users, which could be helpful in dealing with the integration.	Leverage this knowledge to understand and adapt to changes faster. Provide training on advanced features of SAP if required.
The customer is not directly connected with the vendor and expects releases to be on time, regardless of the agile system used.	Establish a communication channel between the customer and the vendor, or act as the intermediary to keep the customer informed about the vendor's schedule and changes. Regularly update the customer on project progress and any potential impacts on the delivery schedule.
The team feels demotivated due to frequent unplanned changes.	Incorporate change management strategies, like preparing the team for potential changes at the beginning of each sprint. Make sure team members feel supported and heard during changes. Recognize and reward adaptability and problem-solving.
There is pressure to align the release cycle with the customer's schedule.	Discuss the possibility of aligning the release cycle with the customer. If feasible, adjust the sprint schedule to

Understanding	Action
	align with the customer's expectations, while maintaining the team's capacity and avoiding burnout.
The team is currently using Scrum, which might not be the most flexible for an environment with frequent changes.	Consider introducing a hybrid model such as Scrumban, which combines Scrum's structure with Kanban's flexibility, to handle the changing nature of work.

Again, it's crucial to keep the lines of communication open and ensure all parties are kept informed of any changes or delays. Also, team morale and motivation is key. Support your team, listen to their concerns, and make them part of the solution.

$ Do you additional suggestions how we should handle customer ad-hoc requests effectively without disturbing too much our sprint implementation?

ChatGPT

Absolutely, here are some additional strategies to handle customer ad-hoc requests without causing too much disruption to your sprint implementation:

1. **Establish Clear Prioritization**: Develop a system to prioritize ad-hoc requests based on their urgency, impact, and effort required. Not all ad-hoc requests need to be addressed immediately, and some might even be included in the backlog for future sprints.
2. **Create a Buffer**: Include a buffer in each sprint to handle ad-hoc requests. This could be a certain percentage of the team's time or a specific number of story points. This will ensure that there is always some capacity to handle unexpected work without disrupting the rest of the planned tasks.
3. **Dedicate a "Firefighter"**: On a rotating basis, assign one team member to handle all ad-hoc tasks for a sprint. This allows the rest of the team to focus on their planned tasks while ensuring ad-hoc tasks are not ignored.

4. **Use a Kanban Board**: Consider using a separate Kanban board to visualize, track, and manage ad-hoc tasks. This allows the team to see the status of these tasks and helps in managing the flow of work.
5. **Frequent Communication**: Keep the lines of communication open with the customer. Regularly update them on the status of their requests and educate them on the impact of ad-hoc requests on the team's productivity and the project's progress.
6. **Review and Learn**: After each sprint, review how well ad-hoc requests were handled. Use this to learn and improve the process for future sprints.

Remember that the goal of agile is to embrace change, but this should not come at the cost of team productivity or morale. It's a balancing act, and finding the right approach for your specific situation might take some time and experimentation.

$ Can you provide updated action plan?

ChatGPT

Sure, I'll update the previous plan considering your new queries and responses:

Understanding	**Action**
Vendor changes happen within a week or two, aligning with our sprints.	Use the weekly Teams meetings to prepare for these changes, updating the team promptly and adjusting the sprint plan accordingly.
The team has end-user experience with SAP.	Utilize this experience to facilitate a smoother integration process. Consider further SAP training if needed.
The customer is disconnected from the vendor's schedule but	Act as the intermediary between the customer and vendor. Update the customer on the vendor's schedule and potential impacts on the project timeline.

Understanding	Action
demands timely releases.	
The team feels demotivated with frequent unplanned changes.	Implement change management strategies. Recognize and reward the team's adaptability and efforts in dealing with changes.
Pressure exists to align with the customer's release cycle.	Assess the feasibility of aligning the release cycle with the customer's schedule. Adjust the sprint plan without overloading the team.
Ad-hoc customer requests are affecting sprint implementation.	Set up a prioritization system for ad-hoc tasks. Include a buffer in sprints for unexpected tasks. Designate a rotating "firefighter" role to tackle ad-hoc requests. Implement a separate Kanban board for better visualization and tracking of ad-hoc tasks. Increase communication frequency with the customer to manage expectations. Review the handling of ad-hoc tasks after each sprint to identify improvements.
The team's current agile method might not suit the project's changing environment.	Consider integrating elements from other agile methodologies, such as Kanban, to create a hybrid approach like Scrumban, which can provide structure and adaptability.

Please note, this updated action plan incorporates the new insights and suggestions on managing ad-hoc requests and aligning with the customer's expectations. The plan's effectiveness should be reviewed regularly and adjusted based on the project's evolving needs.

Model: GPT-4

Change resistance prompt:

$ As an agile coach your task is to improve team way of working to overcome change resistance. The project team has the following challenges: team has been working in water-fall method over five years, scrum or agile way of working is new to the team, senior developers would like to continue the work as they have used to without dailies and retrospectives. Ask the necessary questions to fully understand the challenges and provide plan how to overcome change resistance. Explain your understanding and thinking to me representing the scrum master in the team. Provide output in action plan format.

ChatGPT

Based on your description, it appears that the project team is deeply entrenched in the waterfall method and is resistant to changes in their workflow, particularly the incorporation of Scrum or Agile practices. This reluctance seems most pronounced among senior developers who don't want to include daily standups and retrospectives into their routines.

To ensure a comprehensive understanding, I'd ask a few additional questions:

1. Has there been any previous attempt to adopt Agile methodologies within the team?
2. Are the team members aware of the benefits of Agile methodologies over the Waterfall model?
3. Is there a particular reason why senior developers are opposed to daily standups and retrospectives? Is it due to a perceived disruption in workflow, lack of understanding, or other reasons?
4. How open is the management to the change towards Agile practices?
5. What current issues, if any, does the team face with the waterfall model?

After gathering all the necessary information, here's an action plan:

1. **Educate and Align on Agile Values:** Conduct a series of educational workshops that provide an overview of Agile methodologies, emphasizing their benefits over the waterfall model. These workshops should cover the principles and values of

Agile and Scrum, and why they might help the team deliver better products. Highlight case studies where similar teams successfully transitioned from waterfall to Agile.

2. **Explain the Roles and Responsibilities:** Clearly define the roles in the Agile Scrum framework – Product Owner, Scrum Master, and Development Team – and their responsibilities. Ensure everyone understands their role and how it contributes to the team's success.
3. **Run Simulations or Pilot Projects:** Start with a small pilot project following Agile methodology to provide hands-on experience without disrupting the entire project workflow. This would provide a low-stakes environment for the team to experiment with the new practices.
4. **Introduce Agile Practices Gradually:** Rather than a complete immediate switch, introduce practices incrementally. Begin with stand-ups, ensuring they are short, focused, and valuable. Once comforTable, introduce sprint planning, retrospectives, etc.
5. **Provide Coaching and Support:** As an Agile coach, provide continuous coaching and support throughout the transition. Encourage questions, ensure everyone's voice is heard, and make adjustments as needed.
6. **Collect Feedback and Adapt:** After each sprint, collect feedback from team members on the new process. Discuss what's working and what's not. This allows for continuous improvement and will help to gradually win over those resistant to change.
7. **Showcase Successes:** Regularly showcase improvements and successes achieved through the new practices. Recognize and celebrate these, as it helps to motivate the team and reinforce the value of the new methods.

Remember that change takes time and patience. It's crucial to foster a supportive environment where team members feel safe to experiment and make mistakes. This will enable them to grow more comforTable with Agile practices and see their value over time.

$ Scrum practicalities are difficult to follow. Team does not understand the importance of sprint planning and daily scrums. Team size is five persons plus product owner who is also having scrum master role. As the whole organization moved to use Scrum also this team needs to follow the process. Organization supports scrum masters but not individual team members. Senior developers want buy the new thinking and want to stick in old way of working.

ChatGPT

Understanding these additional details, it's apparent that the challenges aren't just related to the adoption of Scrum, but also rooted in how the organization is providing support during this transition, as well as a lack of understanding about the value and practicalities of Scrum elements. Here is an adjusted action plan to accommodate these realities:

1. **Executive Support:** Work with organizational leadership to ensure that support is not only provided to Scrum Masters but to individual team members as well. This could involve providing training, resources, or time to learn and adapt.
2. **Role Clarification:** Given that the Product Owner is also playing the Scrum Master role, there's a risk of conflicting interests. Consider introducing another team member to take on the Scrum Master role to ensure checks and balances.
3. **Value-driven Understanding:** Conduct specialized training sessions explaining why each element of Scrum exists and its benefits. For instance, illustrate how sprint planning helps anticipate potential issues, divide work evenly, and manage expectations, and how daily scrums provide transparency, quick issue resolution, and team synchronization.
4. **Start with a Simulated Scrum Cycle:** This would include the entire process - Sprint Planning, Daily Scrum, Sprint Review, and Sprint Retrospective - on a small scale project. It's an opportunity for team members to get hands-on experience and understanding of how these activities are interconnected and vital for project success.
5. **Foster an Open Environment:** Encourage senior developers to express their concerns and challenges regarding Scrum. Address their issues directly, showing empathy and offering practical solutions. This will help build trust and openness to new ideas.
6. **Feedback and Adjustment:** After each Scrum cycle, hold a Retrospective to discuss what went well and what didn't. Make adjustments accordingly. Feedback is

vital for continuous improvement and to assure team members that their voices are heard and valued.

7. **Promote Wins:** Regularly celebrate improvements and successes tied directly to Scrum adoption. Showing tangible benefits will gradually convince the resistant senior developers of the value of the new approach.
8. **One-on-One Coaching:** Offer individual coaching sessions to the senior developers, addressing their specific concerns and helping them see the value in the new process. Highlight how Scrum can benefit them personally, not just the team or organization.

Implementing Scrum is a significant shift and can be challenging. However, with careful planning, communication, patience, and persistence, the team can adopt the new methodologies and reap their benefits.

$ Why your original first action plan is so much different than the second one you provided?

ChatGPT

The initial action plan I proposed was based on the information you provided at the beginning, which painted a general picture of a team resistant to change from a waterfall approach to Agile/Scrum. Based on this scenario, the action plan focused on gradual education, introduction of Agile practices, and fostering a supportive environment for change.

However, in your subsequent message, you provided more specific details about the situation. This additional context included:

1. The team's difficulty in understanding the importance of sprint planning and daily scrums.
2. The dual role of the Product Owner also being the Scrum Master.
3. The organization's support focused on Scrum Masters but not individual team members.
4. The particular resistance of the senior developers.

Given these new insights, the action plan needed to be tailored more specifically to address these issues. Hence, the second action plan had an increased focus on role clarification, enhancing organization-wide support, addressing the concerns of the senior developers, and improving the understanding of Scrum activities' value.

This shift between the two plans highlights the importance of understanding the specific context and challenges when proposing a plan for implementing change. The more detailed the information, the more tailored and effective the action plan can be.

Model: GPT-4

Maintaining agile way of working prompt:

$ As a scrum master your task is to maintain agile way of working. The team has the following challenges: Team does not do proper sprint planning, no retrospectives are made, daily scrums do not have value for the team, team is changing the process without any change management. Ask the necessary questions to fully understand the challenges and create coaching plan for maintaining agile way of working. Explain your understanding and provide plan to me representing the project manager. Provide output in notes.

ChatGPT

Understanding of Challenges:

1. Inadequate Sprint Planning: The team doesn't engage in the necessary process of defining a set work for a given period (Sprint). This might cause issues in organizing tasks, managing time effectively, and understanding the scope of the work.
2. No Retrospectives: Retrospectives are critical for an Agile team to improve. The absence of these sessions means missed opportunities to learn from the past Sprints, which could hinder the team's growth and improvement.
3. Unproductive Daily Scrums: The Daily Scrum meetings are meant to synchronize the team's work and plan for the next 24 hours. If they don't provide value, it could lead to ineffective communication and misalignment among team members.

4. Unmanaged Process Changes: The team is making changes to the process without proper change management, leading to possible confusion, reduced productivity, and resistance among team members due to abrupt changes.

Coaching Plan:

1. **Sprint Planning Improvement:**
    - Understand: What is the current method for Sprint planning? Are all team members involved in this process? How are tasks prioritized and assigned?
    - Coach: Show the benefits of proper sprint planning, such as better task management, clear priorities, and improved productivity. Demonstrate how to conduct an effective sprint planning meeting.
    - Implement: Encourage the team to take part in sprint planning meetings, ensuring everyone's input is considered.
2. **Incorporation of Retrospectives:**
    - Understand: Has the team ever conducted a retrospective? If not, what were the reasons behind this? If yes, what were the outcomes?
    - Coach: Explain the importance of retrospectives in continual improvement. Teach them how to conduct a retrospective effectively.
    - Implement: Schedule regular retrospective meetings at the end of each sprint. Ensure they are used constructively to improve the team's methods and performance.
3. **Value-driven Daily Scrums:**
    - Understand: What is currently being discussed in daily scrums? Why do team members feel they don't add value? Are these meetings being held at a convenient time for all members?
    - Coach: Explain the purpose and format of effective daily scrums. Share best practices for making these meetings productive.
    - Implement: Revamp daily scrum meetings based on feedback from the team and new guidelines. Monitor these meetings for effectiveness and adjust as needed.
4. **Change Management:**

  - Understand: What changes have been made recently? Why were these changes made? How have they impacted the team?
  - Coach: Show the importance of proper change management and how to effectively implement changes in the team.
  - Implement: Establish a protocol for introducing changes. Ensure that any new change is communicated well, understood by everyone, and gradually introduced.

Conclusion:

By understanding the current challenges and providing a step-by-step plan for improvement, we can help the team transition to a more effective Agile way of working. The Scrum Master plays a key role in coaching the team, facilitating communication, and promoting Agile values and principles. The proposed coaching plan aims to address all the identified challenges and support the team in their Agile journey.

$ Team does not see the point in sprint planning as senior developers think they can do it individually. Each person already has some role and no crossteam roles are not seen important. I as senior developer do not see any big difference working as we have used to.

ChatGPT

Understanding this new context, it seems that the team is not fully embracing the principles of Agile and Scrum methodologies, particularly the principles of collective decision-making, shared responsibilities, and continuous improvement. This is causing some challenges in sprint planning, role allocation, and perception of Agile practices.

The key here is to address the mindset and encourage collaboration. Here's a revised coaching plan considering these new insights:

Coaching Plan:

1. **Promote Agile Mindset and Scrum Principles:**
   - Understand: What are the team's perceptions and understanding of Agile and Scrum? What are their concerns about these methodologies?

    - Coach: Highlight the benefits of Agile and Scrum, including improved productivity, better product quality, and increased customer satisfaction. Encourage the mindset of collective ownership, shared decision-making, and continuous learning.
    - Implement: Organize regular workshops and training to foster an Agile mindset and clarify misconceptions about Scrum. Create a safe and open environment for discussions about concerns and fears.
2. **Value of Sprint Planning:**
    - Understand: Why do senior developers believe they can plan sprints individually? How does this impact junior developers and the team dynamics?
    - Coach: Emphasize the importance of collective sprint planning, explaining that it provides a shared understanding of work, ensures alignment, and fosters shared responsibility.
    - Implement: Facilitate collaborative sprint planning sessions where all team members contribute. Encourage senior developers to mentor junior developers during these sessions.
3. **Cross-Functional Roles and Shared Responsibilities:**
    - Understand: Why are cross-team roles not seen as important? How does this impact the team's flexibility and resilience to change?
    - Coach: Explain the advantages of cross-functional roles, including increased team flexibility, reduced bottlenecks, and improved team learning.
    - Implement: Introduce practices that encourage cross-functionality, like pair programming or rotation of roles. Monitor and highlight the positive impact of these practices.
4. **Engagement of Senior Developers:**
    - Understand: What are the senior developer's main concerns with the Agile way of working? How can we address these concerns to enhance their engagement?
    - Coach: Discuss the value and benefits of Agile from a senior developer’s perspective, focusing on elements like improved predictability, shared responsibility, and sustainable work pace.
    - Implement: Encourage senior developers to take active roles in Agile practices. Their engagement could influence other team members positively.

Remember, changing mindsets and behaviors takes time and patience. Consistent encouragement, active listening, and reinforcement of benefits will slowly but surely lead the team towards the Agile way of working.

$ Can you still elaborate how I can maintain the agile way of working without reverting back to the old way of working?

ChatGPT

Certainly, maintaining Agile practices requires a persistent focus on Agile principles and a commitment to continuous improvement. It's important to facilitate a culture shift that encourages transparency, inspection, and adaptation. Here's a more specific plan:

Sustaining Agile Practices:

1. **Consistent Training and Reinforcement:**
    - Reinforce the Agile mindset and the benefits of its practices in all meetings and discussions. Regular training sessions, workshops, and seminars can be beneficial to keep everyone on the same page. Remember to keep the focus on principles and values, not just practices and tools.
2. **Leadership Role Modeling:**
    - As the Scrum Master, exemplify the Agile values and principles in your daily interactions and decision-making. Encourage other leaders within the team, like senior developers, to do the same. This can help to influence the rest of the team to adopt and maintain Agile practices.
3. **Fostering a Culture of Collaboration and Shared Responsibility:**
    - Foster a culture where all team members take collective ownership of the project. This can be achieved through practices such as pair programming, shared decision-making in sprint planning, and rotation of roles. Over time, this encourages cross-functionality and shared responsibility, making it less likely to revert to old ways of working.
4. **Regular Feedback and Continuous Improvement:**
    - Conduct regular retrospectives to discuss what's working and what's not. Encourage the team to share their thoughts and concerns. Make necessary

        adjustments based on feedback to ensure Agile practices are beneficial and not just a set of rituals.
5. **Metrics and Transparency:**
    - Implement Agile metrics that make sense for your team, such as velocity, sprint burndown, or cumulative flow diagrams. This can help the team see the value of Agile practices in real-time and make it easier to stick to them.
6. **Acknowledge and Celebrate Success:**
    - It's important to acknowledge the team's hard work and celebrate when milestones are achieved or when improvements are noticed. This can boost morale and incentivize the maintenance of Agile practices.

Remember, Agile is more of a mindset than a set of practices. It's about being adaptive, customer-focused, and team-oriented. By fostering this mindset, you'll help the team maintain the Agile way of working and resist the urge to revert to old practices.

Model: GPT-4

Effort estimation prompt:

$ As a senior developer your task is to create an effort estimation. The user story has the following requirements: User needs to login to the system, she needs to enter her email address and password, password needs to be unique and is stored in to nosql db, password needs to be change every 30 days and cannot be reused. Ask the necessary questions to fully understand the requirements and create effort estimation and required tasks. Explain your understanding and provide effort estimation to me representing the scrum master. Provide output in Table containing tasks and effort needed.

ChatGPT

Sure, I will need to clarify a few things in order to fully understand the requirements:

1. Are there any specific platforms or languages that the login system needs to be developed for?
2. Do we already have an existing user database, and if so, is it the same NoSQL database that will be used for storing the password?

3. Are there any particular standards or specifications for the unique password? E.g., minimum length, combination of special characters, letters and numbers, etc.
4. When you mention "password needs to be unique", does that mean unique across all users or unique to the user's previous passwords?
5. When the password changes every 30 days, is the user required to change it manually or is it an automatic process?
6. What kind of notification should the user receive for the password change, and when should they receive it?
7. What happens if a user tries to reuse a password? What kind of error or message should they receive?

Assuming that I have the answers to these questions, here's an example effort estimation:

Task	**Description**	**Effort**
Task 1	Research and Planning	5 hours
Task 2	Set up user database in NoSQL (assuming this is a new database)	8 hours
Task 3	Develop login system (email + password input validation)	15 hours
Task 4	Create password uniqueness validation	10 hours
Task 5	Develop system to enforce password change every 30 days	20 hours
Task 6	Develop notification system for password change	10 hours
Task 7	Develop system to prevent password reuse	10 hours
Task 8	Testing and bug fixes	10 hours
Task 9	Deployment and Documentation	5 hours
Total		**93 hours**

This Table presents a rough estimate. The actual effort may vary based on the specific technology stack used, the complexity of the existing system, and the actual requirements from the client. Furthermore, this estimation does not include time for meetings, requirement changes, or other possible work interruptions. It's always a good idea to add a buffer to the estimate to account for these factors.

$ We use existing AA in backend which can be conFigured for our purpose. So the topic is actually only designing the web UI. Password has minimum length of 12 characters and should have at least two special characters and two capital letters. Uniqueness is stored in the backend so nobody can use password used once. Email verification is used. All passwords since the beginning are stored. NoSql will be Redis as it is fast and scalable. We use three Redis nosql nodes. Passwords are hashed using HMAC and salting is also needed.

ChatGPT

Thank you for the additional details. Given that the backend processes like email verification, password storage, uniqueness checks, hashing and salting are already being handled, we're primarily dealing with designing and developing the UI and the corresponding front-end validations. Here's a revised effort estimation:

Task	Description	Effort
Task 1	Research and Planning	3 hours
Task 2	Designing UI for login page and password change page	6 hours
Task 3	Implementing UI for login page and password change page	8 hours
Task 4	Front-end password validation (length, special characters, capital letters)	8 hours

Task	**Description**	**Effort**
Task 5	Implement mechanism to communicate with backend for email verification	6 hours
Task 6	Create front-end notifications/messages for user guidance (password change, reused password, etc.)	6 hours
Task 7	Integration with existing backend authentication/authorization	10 hours
Task 8	Testing (unit, integration, and UI/UX testing) and bug fixes	10 hours
Task 9	Deployment and Documentation	4 hours
Total		**61 hours**

As before, these estimates could vary based on the specific requirements, the existing system, the UI/UX complexity, the technology stack used, and the project management methodology applied. A contingency buffer should still be considered for unexpected issues or requirements changes.

$ On what basis work is estimated in hours?

ChatGPT

Effort estimation in software development is based on a variety of factors, and different organizations might use different methodologies to estimate the work required. However, some of the common factors considered when estimating work in hours include:

1. **Task Complexity**: The complexity of a task is a major factor in estimation. Complex tasks require more time to complete than simple ones. For example, developing a custom algorithm might take more time than writing a CRUD operation for a database.
2. **Past Experience**: Developers often base their estimates on their past experience with similar tasks. If a developer has previously spent 5 hours developing a similar feature, they might estimate a similar amount of time for the current task.
3. **Developer Skill Level**: Different developers have different levels of expertise. A less experienced developer might need more time to complete a task than a more experienced one.
4. **Available Resources**: The availability of resources, including software, hardware, and other necessary tools, can also affect the estimated time. If a developer has to spend time setting up a development environment or searching for necessary resources, this can add to the time required for a task.
5. **Interruptions and Unforeseen Challenges**: Developers often add a buffer to their estimates to account for potential interruptions or unforeseen challenges that might arise during development. This might include issues like bugs, changes in requirements, or other unexpected problems.
6. **Review, Testing, and Debugging Time**: It's also necessary to estimate the time required to review the code, test it, and debug any issues. This can often be a significant part of the total time spent on a task.

Estimating work in hours is not an exact science, and it often involves a degree of educated guesswork. The goal is to provide a reasonable estimate that allows for planning and scheduling of the work, not to predict the exact amount of time a task will take. Over time, as a team works together and gains more experience, their estimates tend to become more accurate.

$ Why you give estimations in hours and not e.g. in story points?

ChatGPT

Estimations can indeed be done in many ways, and story points are a common alternative to hours. Both methods have their own strengths and can be applied depending on the context.

Story points are a measure of effort that incorporates complexity, risk, and repetition. They are a unit of measure for expressing an estimate of the overall effort required to fully implement a product backlog item or any other piece of work. Story points enable a team to estimate work in a way that helps to hide minor discrepancies in individual productivity and work habits.

Estimating in story points rather than in hours can have some advantages:

1. They're relative to each other, not an absolute measure of time, so they can account for the unknowns and complexity better.
2. They're more effective for long-term planning and forecasting.
3. They help prevent micro-management as they avoid giving a false impression of precision.

However, the method of estimating in hours can be helpful in certain contexts:

1. They can be easier to understand for people new to Agile, as they are a traditional way of estimating work.
2. They can feel more "tangible" and can be a good way to estimate smaller, well-defined tasks.
3. Estimating in hours can also be more suiTable for sprint planning when the team needs to understand the granular day-by-day work involved.

In this particular scenario, since you've asked for an estimation and we're focusing on the specific tasks required to implement a feature, I provided an estimation in hours as it is a straightforward and commonly understood method. However, depending on the actual development process and team preferences, story points could be used as well.

Model: GPT-4

Technical knowledge analysis prompt:

$ As an engineering manager your task is to estimate technical knowledge in team to support effort estimation. The project has the following technical objectives: Backend is developed using Java and SQL database, system is running on top of RedHat Linux, we need some sort of DevOps, we should need about 3-4 man months to do the project, existing code base exists so only new functionality need to be developed. Ask the necessary questions to fully understand the technical objectives and create plan how team can fulfill the needed technical skills. Explain your understanding and provide plan for me representing the project manager. Provide output in memo format.

ChatGPT

Subject: Technical Skills Assessment and Development Plan for Project

Dear Project Manager,

Following the project briefing, I have comprehensively analyzed the required technical objectives. The project entails backend development using Java and an SQL database, implementation on a RedHat Linux system, and incorporates DevOps principles. Moreover, it is vital to note that an existing code base is present, and our focus will primarily be on introducing new functionalities.

Before devising a development plan, we must confirm certain aspects to ensure our team can fulfill the required technical skills:

1. Java: What is the specific version of Java that we are using for backend development? How extensive is the Java experience among our team? What libraries and frameworks are we using, and are our team members familiar with them?
2. SQL Database: What specific SQL database are we using? How well-versed are our team members in creating, querying, and managing the chosen SQL database?
3. RedHat Linux: How comforTable are our team members in running and troubleshooting a system on RedHat Linux?
4. DevOps: What DevOps tools and methodologies are we planning to use? Do our team members have experience with these tools?

5. Existing Code Base: Is the existing code base well-documented? Do our team members understand the existing code base, its architecture, and design patterns?
6. Project Complexity: How complex is the new functionality? What is the impact on the existing system?

Once we have the answers to these questions, we can outline a skill development plan:

1. Training and Workshops: If gaps in Java, SQL database, RedHat Linux, or DevOps skills are identified, we can organize training sessions or workshops to improve the team's competency. This will provide an in-depth understanding and hands-on experience.
2. Knowledge Transfer Sessions: If some team members are already skilled in the requisite areas, we can conduct internal knowledge transfer sessions. This not only promotes skill development but also fosters team bonding.
3. External Courses: If necessary, we can encourage team members to enroll in external courses for certifications in Java, SQL, RedHat Linux, or DevOps. This can be done through platforms like Coursera, Udemy, etc.
4. Code Reviews: Encourage regular code reviews to ensure understanding and adherence to best practices. This will improve our team’s ability to work with the existing codebase.
5. Assign Mentors: For new functionalities, assigning a mentor for every team member can be beneficial. The mentor can guide them through the complexity of the new functionality and its impact on the existing system.
6. Mock Projects: To assess progress and readiness, we can assign small mock projects encompassing the necessary skills.
7. Pair Programming: Implement pair programming sessions where team members collaborate and learn from each other's skills.

Finally, it's essential to monitor the progress continuously and adjust the plan based on feedback and results.

Please provide the necessary details at your earliest convenience, so we can move forward with the skills assessment and development plan accordingly.

Best Regards,

[Your Name] Engineering Manager

The integration is using CSV, XML and ASCII files between the systems. These are transferred using sftp. Team has two software engineers and they know the basics but not have done this before. Project should be ready in three months. We don't have any training resources and team members need to browse internet and ask from Google.

ChatGPT

Subject: Revised Technical Skills Assessment and Development Plan for Project

Dear Project Manager,

Thank you for providing further details about the project and our team’s competencies. Considering our integration will use CSV, XML, and ASCII files, and files will be transferred via sftp, and the current knowledge level of our team, here is the revised development plan:

1. Self-Learning and Research:

Given that we don't have specific training resources available, encourage team members to utilize the vast pool of knowledge on the internet. Websites such as Stack Overflow, GitHub, official documentation, and various blogs and tutorials are rich resources for learning and understanding concepts.

2. Skill Areas to Focus on:

- File handling in Java: Understanding how to read and write CSV, XML, and ASCII files. Also, they should focus on understanding error handling and data validation for these files.
- Understanding sftp: This involves learning how to set up sftp connections, transfer files securely, handle potential connection errors, etc.

- SQL: Focusing on how to handle database interactions efficiently, including how to correctly import data from CSV, XML, and ASCII files.

3. Collaborative Learning:

- Pair programming: This encourages team members to work together, fostering collective problem-solving and knowledge sharing.
- Regular meetings: Team members can share what they learned, discuss challenges they faced, and brainstorm solutions. This continuous exchange of information helps everyone learn faster and promotes problem-solving skills.

4. Practical Application:

- Mock Projects: Develop small projects or modules to simulate actual project scenarios. This allows them to apply what they've learned and gain practical experience.
- Code Reviews: Regular reviews of code written for the mock projects will provide valuable feedback on their learning progress and areas they might need to focus more on.

5. Time Management:

Considering the project timeline is three months, the learning phase should be compressed as much as possible without compromising the quality of learning. Proposed timeline:

- Weeks 1-2: Dedicated to learning file handling and sftp in Java, and SQL database interactions.
- Weeks 3-4: Focus on practical applications, mock projects, and learning from the feedback received during code reviews.
- Remaining time: Implementing the project, including additional learning as necessary.

Please let me know if you have any questions or if there are additional areas we should focus on.

Best Regards,

[Your Name] Engineering Manager

$ How I could use information you provided in the next sprint plannings to estimate needed time?

ChatGPT

Subject: Utilizing Technical Skill Development Plan in Sprint Planning

Dear Project Manager,

In Agile project management, a sprint planning session defines what can be delivered in the upcoming sprint and how the work will be achieved. The technical skill development plan I provided can be instrumental in determining the team's velocity (the amount of work the team can handle in one sprint) and task allocation, both of which will influence the time estimation.

Here's how you can utilize this information:

1. Assess Current Skill Levels:

The skill development plan provides an overview of the team's current competencies and areas of improvement. This information can guide how tasks are allocated in the initial sprints.

2. Allocate Time for Learning:

The first few sprints will need to incorporate time for the team members to enhance their skills. This could include self-study, online courses, and pair programming sessions. This dedicated learning time should be factored into the sprint planning and will likely reduce the team's initial velocity.

3. Gradually Increase Workload:

As the team's skills improve over time, you can gradually increase the workload in each sprint. This process should be closely monitored and adjusted based on the team's feedback and overall performance.

4. Monitor and Adapt:

Keep track of the time taken for each task during the sprints. If certain tasks are consistently taking longer than estimated, it may indicate that further training is required in that area.

5. Factor in Time for Code Reviews:

The plan emphasizes regular code reviews to ensure understanding and adherence to best practices. This should also be included in the sprint planning.

6. Use the Mock Projects to Improve Estimates:

The mock projects' performance will give you a baseline for estimating similar tasks in the actual project. This 'real-world' experience can be invaluable in improving your time estimates for future sprints.

7. Retrospective Meetings:

Regular retrospective meetings at the end of each sprint will help the team identify what went well and what didn't. This continuous learning will feed into future sprint planning, helping to make your time estimates more accurate.

Remember, the goal is to continuously adapt and improve, using each sprint as a learning opportunity. I'm confident that with careful planning and monitoring, our team can meet the project's technical objectives successfully and efficiently.

Best Regards,

[Your Name] Engineering Manager

I forgot to mention Java version we use is 17 and db is Postgresql

ChatGPT

Subject: Revised Technical Skills Development Plan Based on New Information

Dear Project Manager,

Thank you for sharing the specific versions of Java and the database system we'll be using for our project. Here's an adjusted plan taking into account Java 17 and PostgreSQL:

1. Self-Learning and Research:

- Java 17: The team members need to familiarize themselves with the features, enhancements, and any deprecated APIs in Java 17 compared to the version they are currently familiar with. The official Oracle documentation, online tutorials, and community forums will be excellent resources for this.
- PostgreSQL: Team members should focus on understanding the specifics of PostgreSQL, its syntax, functions, and best practices for efficiency and performance. Official PostgreSQL documentation and tutorials are rich resources for the same.

2. Practical Application:

- Java 17: Encourage team members to use new Java 17 features in their code. This will help them better understand these features and how they can be applied to improve code efficiency and readability.
- PostgreSQL: Practical exercises on creating, managing, and querying PostgreSQL databases will provide team members hands-on experience. They should also be trained in importing data from CSV, XML, and ASCII files into PostgreSQL databases.

All other aspects of the original plan remain the same. Please provide any other specific details about the project at your earliest convenience to help the team prepare more effectively.

Best Regards,

[Your Name] Engineering Manager

$ Can you provide the full memo?

ChatGPT

Subject: Comprehensive Technical Skills Assessment and Development Plan for Project

Dear Project Manager,

Following our discussions regarding the upcoming project, the technical requirements, and the skillset of our team, I have refined our Technical Skills Assessment and Development Plan. The project involves backend development using Java 17 and a PostgreSQL database. The integration will employ CSV, XML, and ASCII files, transferred using sftp. The system will run on RedHat Linux, incorporating DevOps principles.

Our team consists of two software engineers with basic skills but no direct experience with the specifics of this project. Therefore, I've outlined the following steps for our strategy to meet the project's technical objectives within the three-month timeline:

1. Self-Learning and Research:

Given that we don't have specific training resources available, we'll encourage our team members to leverage the vast pool of knowledge available on the internet. Websites such as Stack Overflow, GitHub, official documentation, and various blogs and tutorials are rich resources for learning and understanding concepts.

Specific Skill Areas to Focus on:

- Java 17: Team members need to familiarize themselves with new features, enhancements, and deprecated APIs in Java 17.
- PostgreSQL: Understanding specifics of PostgreSQL, its syntax, functions, and best practices for efficiency and performance.
- File handling in Java: Reading and writing CSV, XML, and ASCII files, error handling, and data validation.

- sftp: Setting up sftp connections, secure file transfers, and handling potential connection errors.

2. Collaborative Learning:

- Pair programming: Encourages team members to work together, fostering collective problem-solving and knowledge sharing.
- Regular meetings: Team members share what they've learned, discuss challenges, and brainstorm solutions. This continuous exchange of information helps everyone learn faster and promotes problem-solving skills.

3. Practical Application:

- Mock Projects: Create small projects or modules to simulate actual project scenarios. This will allow team members to apply what they've learned and gain practical experience.
- Code Reviews: Regular reviews of code written for the mock projects will provide valuable feedback on learning progress and areas requiring more focus.

4. Time Management:

Considering the project timeline is three months, the learning phase should be compressed as much as possible without compromising the quality of learning. The proposed timeline:

- Weeks 1-2: Dedicated to learning Java 17, PostgreSQL, file handling, and sftp.
- Weeks 3-4: Focus on practical applications, mock projects, and learning from feedback received during code reviews.
- Remaining time: Implementing the project, including additional learning as necessary.

As we progress with the plan, I recommend using the team's skill development and learning curve to estimate the needed time during the sprint planning sessions. Keep in mind the initial few sprints might have a lower velocity due to the learning phase, but it should increase as the team's skills improve. Regular monitoring and adjustments based on feedback and results will ensure we remain on track to meet our project objectives.

Looking forward to your feedback on this plan and further details if any.

Best Regards,

[Your Name] Engineering Manager

APPENDIX D: PROMPT CHEAT SHEET

As a solution to provide help to each agile project pain point a cheat sheet is provided containing contextual and customizable prompt patterns. Statements in braces specifies the parameters used in the patterns. General and customizable prompt pattern is described in Table 15. Intention is that items written in bold font can be modified in the prompt pattern.

Table 15. General contextual prompt pattern

As a **[roleA**] your task is to **[do-what]**. The **[what]** must fulfill the following **[constraints]**: ***c1, c2, c3…*** Ask the necessary questions to fully understand the **[constraints].** Based on my answers **[define-outcome]**. Explain your understanding and **[requested-information]** to me representing the **[roleB]**. Provide output in [format-specification].

Individual prompt patterns are designed per agile project pain point and are named accordingly in the Table 16. Parts written in braces are intended to be modified according to the context of the pain point.

Table 16. Prompt pattern cheat sheet for agile pain points

Requirements creation prompt	

	`As a [roleA] your task is to create a requirement specification.`
	`The requirement specification must fulfil the following constraints: [c1, c2, c3…].`
	`Ask the necessary questions to fully understand the constraints.`
	`Based on my answers create the requirement specification.`
	`Explain your understanding and construct your proposal to me representing the [roleB].`
	`Provide output in [format].`

Steering group prompt	

	As a [roleA] your task is to fully understand agile principles, business objectives and give feedback about the project. The project has the following objectives: [o1, o2, o3…]. Ask the necessary questions to fully understand the objectives and to be able to understand the project status and way of working. Explain your understanding and provide necessary actions for me representing the [roleB]. Provide output in [format].
Role clarification prompt	
	As a [roleA] your task is to specify clear roles for the team. The project team has the following constraints: [c1, c2, c3…]. Ask all necessary questions to fully understand the project and be able to create necessary role specifications. Explain your understanding and provide solution to me representing the [roleB]. Provide output in [format].
Redundancy analysis prompt	
	As a [roleA] your task is to analyze redundancy in agile team skills. The project team has the following persons: [p1, p2, p3…]. Responsibilities of the team are: [r1, r2, r3…]. Ask the necessary questions to fully understand the task and create analysis of redundancy gaps in the team. Explain your understanding and provide solution to me representing the [roleB]. Provide output in [format].
Agile process coaching prompt	
	As a [roleA] your task is to coach agile processes. The project has the following objectives: [o1, o2, o3…]. Ask the necessary questions to fully understand the objectives and create agile coaching plan to the team. Explain your understanding and show me the plan representing the [roleB].

	Provide output in [format].
Adaptability management prompt	
	As a [roleA] your task is to coach team to adapt and motivate in occurring changes. The project has the following objectives: [o1, o2, o3…]. Ask the necessary questions to fully understand the objectives and create a plan how to adapt to the changes. Explain your understanding and provide the actions to me representing the [roleB]. Provide output in [format].
Change resistance prompt	
	As a [roleA] your task is to improve team to way of working to overcome change resistance. The project team has the following challenges: [c1, c2, c3…]. Ask the necessary questions to fully understand the challenges and provide plan how to overcome change resistance. Explain your understanding and thinking to me representing the [roleB]. Provide output in [format].
Agile way of working prompt	
	As a [roleA] your task is to coach team to maintain agile way of working. The project has the following challenges: [c1, c2, c3…]. Ask the necessary questions to fully understand the challenges and create coaching plan to maintain agile way of working. Explain your understanding and provide plan to me representing the [roleB]. Provide output in [format].
Effort estimation prompt	
	As a [roleA] your task is to create an effort estimation. The user story has the following requirements: [r1, r2, r3…]. Ask the necessary questions to fully understand the requirements and create effort estimation and required tasks.

	Explain your understanding and provide effort estimation to me representing the [roleB]. Provide output in [format options].
Technical knowledge analysis prompt	
	As a [roleA] your task is to *estimate technical knowledge in team to support effort estimation*. The project has the following *technical objectives*: [o1, o2, o3…]. Ask the necessary questions to fully understand the *technical objectives* and create plan how team can fulfill the needed technical skills. Explain your understanding and provide plan for me representing the [roleB]. Provide output in [format].