Turkish Journal of Engineering – 2025, 9(1), 152-178

Turkish Journal of Engineering

https://dergipark.org.tr/en/pub/tuje

e-ISSN 2587-1366

A formal and integrated approach to engineering machine learning processes: A method base for project management

Murat Paşa Uysal[*1]

[1]*Baskent University, Department of MIS, Turkey, mpuysal@baskent.edu.tr*

Cite this study: Uysal, M.P. (2025). A formal and integrated approach to engineering machine learning processes: A method base for project management. Turkish Journal of Engineering, 9 1), 152-178.

https://doi.org/10.31127/tuje.1530981

Keywords
Machine learning
Project management
Method engineering
Process algebra
Healthcare

Research Article

Received:10.08.2024
Revised:15.09.2024
Accepted:16.09.2024
Published:20.01.2025

Abstract

Enhancing project management (PM) for machine learning (ML) requires structured acquisition and application of PM knowledge. However, significant differences exist between managing ML-enabled software products (MLESP) and traditional software products (TSP). In modern tool-centric ML environments, creating a method base to support team learning and knowledge management is challenging. Studies also show that a "one-size-fits-all" approach to PM can fail to meet diverse team and organizational requirements. Indeed, the main challenge is capturing, storing, and reusing tacit knowledge on PM methods, processes, tasks, and tools for ML. The experimental, data-driven nature of ML may often lead to ad hoc processes, complicating integration with traditional software lifecycles. Therefore, tailoring a PM method for MLESP becomes critical. This study uses a mixed research approach combining Design Science Research (DSR), PM, Method Engineering (ME), and Process Algebra (PA). Key outputs include an ME framework for PM, a method base for ML, and a hybrid ML PM method tailored for Baskent University Hospital Ankara (BUHA). A use case-based scenario analysis technique validated the requirements phase of the hybrid ML PM method in the context of BUHA. The proposed approach can offer comprehensive, yet pragmatic and adaptable solutions as it blends the strengths of ML, PM, ME, and PA knowledge domains. Moreover, PA contributes formal and mathematical foundations for specifying and validating PM methods and tailoring processes. This study has the potential to contribute not only to ML PM and BUHA but also to advancing process management within the mission and safety-critical domains like healthcare.

1. Introduction

The rapid advancement of artificial intelligence (AI), as in machine learning (ML) and deep learning, continues to reshape our community and business landscape, which requires new ways of working environments and business models. Despite the popularity and remarkable success stories associated with ML applications, it's also essential to acknowledge some potential challenges. There have been instances of overestimating business objectives, unrealistic expectations, and a high risk of failure in ML projects. Selecting a feasible business case, aligning business processes with AI processes, and fostering an appropriate organizational project management (PM) culture also arise as important factors [1]. However, most studies tend to focus on technical aspects of ML such as data processing, model development, and deployment [2]. Little emphasis is given to the software engineering (SE) aspects, especially tailoring, or adopting a PM method for ML-enabled software products (MLESP) [3]. Therefore, successfully transitioning an organization toward AI and ML capability involves careful consideration of PM knowledge management.

The first research argument of this study lies in the belief that improving SE practices and PM capability for ML requires the systematic acquisition, structuring, and application of PM knowledge [4]. The key factors affecting the tailoring or customization of PM methods include context-specific requirements (such as the organization and project type), team-specific factors, compliance with standards, and, most importantly, the retention and transfer of PM knowledge. Therefore, building a formal method base or repository for customizing PM methods can significantly contribute to creating a rich knowledge base [5]. Thus, tailored PM methods can enhance efficiency and effectiveness as well as optimize resource utilization, which ensures processes are aligned with organizational and project goals. Project stakeholders' satisfaction can be improved through better communication, engagement, and increased adaptability by allowing PM methods to be scaled and customized. Therefore, a well-maintained PM knowledge can not only foster continuous improvement and adaptation but also ensure that PM practices can evolve with emerging developments in the methods, techniques, and tools for ML [6].

The second research argument is that there are substantial differences between MLESP and traditional software products (TSP) when adopting or tailoring PM methods [7]. Several critical factors need to be considered regarding the distinct differences from TSP. For example, the requirements of TSP can be well-defined and may remain relatively stable while the requirements of MLESP can change because of the experimental nature of ML [3]. Stakeholders can easily provide the requirements that will guide the development processes of TSP, validation and verification of these requirements. As for MLESP, data and model performance requirements drive the development process and therefore, they involve continuous validation, verification, and evaluation of ML models [8]. Well-established PM methods such as plan-driven and agile, can be adopted for TSP. However, data processing forms the integral parts of the development process for MLESP. While iterations are often based on adding or refining the TSP features, the iterations of MLESP are focused on improving model performance through data refinement, feature engineering and model testing. Once a TSP is deployed, it usually requires periodic updates or bug fixes. However, MLESP needs continuous monitoring, retraining, or updating of the ML models to adapt to changing data patterns. Most of the PM methods for TSP have defined team roles, but for MLESP, new roles may come into play in addition to the data scientist and SE roles [2].

1.1. Motivation

In essence, customizing or tailoring a PM method for the specific demands of MLESP emerges as a significant challenge [9]. Due to the black box, data-centric, and experimental nature, ML processes can be easily evolved into ad hoc processes. The studies and industrial applications indicate that a "one-size-fits-all approach" for PM can be impractical or can fall short of the expectations of teams and organizations [10]. Moreover, SE and PM best practices may not be easily tailored to different project types and application domains, primarily because their method parts may not be easily extracted, adapted, or combined [11]. Therefore, an effective PM environment for MLESP should be able to encompass and harmonize the ML development and software development life cycle processes [12].

In contemporary tool-centric PM environments, it is not easy to design and build a method base for PM activities to enable team learning [5,11]. The knowledge required for tailoring a method may not be easily shared or disseminated among team members. The guidelines for customizing PM processes are presented in informal ways, textual or visual formats, lacking formal and common ground for engineering the PM methods [13]. The main challenge lies in how to elicit, store, and reuse tacit knowledge related to the methods, processes, tasks, and tools of PM [4]. Therefore, it is possible to state that there is a need for a method engineering (ME) framework that not only illustrates what can be accomplished through a given PM method but also provides insights into tailoring previously applied methods [11].

The Software Engineering for Machine Learning Applications International Symposium (SEMLA) was an important attempt to foster collaboration between practitioners and researchers to explore the challenges and implications at the intersected areas of SE and ML [2]. Consequently, two fundamental questions emerged: (a) "How should software development teams integrate the AI model lifecycle (training, testing, deploying, evolving, and so on) into their software process?" and (b) "What new roles, artifacts, and activities come into play, and how do they tie into existing agile processes?". Therefore, we argue that the questions discussed in SEMLA require the employment of the methods or techniques of two disciplines: (a) ME and (b) process management.

In the context of ME, we can establish a similarity between a method user and an end-user of a software product. Both seek effective tailored solutions that meet their specific requirements. The decision to either construct a new method from the ground up or tailor an existing one hinges on situational factors that could pertain to the organizational, team, or project levels [11]. Thus, ME can be instrumental in addressing the challenges posed by the diverse and dynamic nature of the MLESP, system, and software development life cycle processes. As for process management, ME can provide a systematic approach for improving and optimizing the PM processes to enhance efficiency, effectiveness, and adaptability. However, process management also needs to leverage engineering and formal approach to connect the diverse domain processes of SE, ML, and PM. Therefore, Process Algebra (PA) can connect these knowledge domains and provide the formal ground needed to model, analyze, and verify the PM process models with high precision and reliability.

Consequently, the existing literature falls short of providing solutions to the problems given above, which also leaves engineering the PM processes for MLESP relatively neglected. Therefore, our research study seeks to address this problem. We believe that standardized representation, modeling, enactment, and deployment of reusable PM methods in a method base can pave the way for more adaptable PM processes for MLESP [7]. The subsequent parts of this paper include a review of related work, theoretical foundations, research method, presentation of the research outputs, discussion, and conclusion sections respectively.

2. Related Work

In the context of SE challenges associated with ML, several literature review studies have contributed different insights. Kumeno [14] conducted the first review mapping the SE challenges for ML to the SE knowledge areas defined by the Software Engineering Body of Knowledge (SWEBOK) [15]. Lwakatare et al. [16] focus on large-scale ML systems in real-world industrial settings. Nascimento et al. [17] highlight various categories, including test, software quality, data, management, model development, and PM in the context of SE challenges for ML. Giray [3] also takes an SE perspective on ML system engineering and presents findings from an extensive systematic literature review on the state-of-the-art and SE challenges for ML. Key issues identified include forming cohesive teams, improving the requirements process, tailoring development approaches, and assessing the SE process [18].

Some of the research reports the case studies of companies engaged in ML projects [19, 20]. They identify ML-specific technical problems, including issues related to data processing, algorithm usage, incomplete tests, and evaluations. ML projects often face the unique challenge of not being able to conduct detailed requirement analysis and specification from the outset. The black-box nature of ML algorithms also makes it challenging to provide explanations to both technical and non-technical stakeholders, leaving questions about "what is possible and what is not" [21]. ML processes tend to be task-focused, and the traditional or stepwise approaches can complicate planning and coordination tasks. However, ML project managers may prefer PM methods reminiscent of traditional, plan-driven, waterfall methods, such as Cross-Industry Standard Process for Data Mining (CRISP-DM). The inflexibility of fixed-length sprints in PM methods like Scrum can lead to the inclusion of unrealistic sprint backlog items. For example, data analysis or model development tasks may require varying amounts of time and effort. Therefore, the adaptability of sprint lengths becomes critical, allowing teams to modify them according to the specific requirements of ML experimentation processes [7].

Effective coordination between teams and stakeholders stands out as another issue in ML projects. These projects often rely heavily on the knowledge and technical expertise of team members such as ML engineers and data scientists [11]. Team members need to possess expertise in various domains, encompassing data processing, statistics, algorithms, and application development. Teams utilizing an immature ML PM method may find themselves overly dependent on senior data scientists or ML practitioners. In a similar context, Saltz et al. [7] compare data science teams using Scrum, Kanban, CRISP-DM, and baseline PM methods. Their findings highlight that CRISP-DM excels in requirements specification and PM processes but faces delays in analytics, modeling, and coding. Scrum teams can encounter challenges in understanding client and data requirements, leading to estimation difficulties or reduced confidence in sprint task completion.

Some researchers point out the need for establishing a consensus on the critical success factors and key performance indicators specific to ML and agile analytics projects [22-23]. Amershi et al. [11] conducted a case study on the SE challenges encountered within ML projects at Microsoft. They note that integration of ML and SE modules introduces a higher level of difficulty and intricacy. The automation of ML pipeline processes becomes a shared concern among ML teams. Therefore, the dynamic and complex context surrounding the development of large-scale MLESP differs substantially from traditional or agile software development contexts. Therefore, there is a growing need to integrate ML workflow management into SE processes to address the adaptability, scalability, and development challenges [11].

The importance of hybrid PM methods for complex software products like MLESP is underscored by some studies. For example, Kuhrmann et al. [24] provide insights into the practical implementation of hybrid software and system development, combining waterfall and Scrum methods, in real-world projects. They highlight effective communication, stakeholder involvement, and skilled PM, contributing to project success in hybrid development. Therefore, Situational Method Engineering (SME) offers an architecture model and provides insights into the development of hybrid PM methods [25]. Papadakis and Tsironis's research [26] involves a review of PM methods, specifically plan-driven, agile, and hybrid techniques. They point out a growing interest in hybrid PM approaches, which integrate practices from plan-driven and agile approaches. Conforto and Amaral [27] introduce a hybrid PM framework that blends agile PM and the Stage-Gate model, specifically tailored for technology-based companies. They report the benefits of flexibility, adaptability, and innovation alignment while acknowledging the challenges of implementation. Zasa et al. [28] aim to understand the dynamics and challenges, and then propose strategies for corrective actions, and for the coexistence of agile, Stage-Gate, and hybrid PM in organizations. The coexistence can be driven by the recognition that different phases within projects may require varying levels of adaptability. Sithambaram et al. [29] also report the issues and challenges impacting the successful management of agile-hybrid projects by using a grounded theory approach. They suggest that effective management of agile-hybrid projects requires a holistic approach that addresses challenges in multiple categories, such as organizational support, alignment with business goals, teamwork, education, and skill development. Azenha et al. [30] explore the role and characteristics of hybrid approaches to PM for technology-based products and services. They emphasize that hybrid methods can allow for a balance between structured planning and the flexibility required for risk management, innovation, and rapid change.

3. Theoretical Foundations

3.1. Machine learning

ML is composed of a range of mathematical, statistical, data science, and computer science methods, processes, and techniques. It uses data and complex algorithms to create learning models that determine patterns within training and test data. The main objective of an ML project is to generate knowledge that informs better decision-making and provides valuable insights derived from this knowledge. As can be seen from Figure 1, the ML process starts with the problem definition, which involves clear identification and articulation of a business problem that the ML project aims to solve. Data acquisition and processing includes gathering, preparing, and transforming data into a suitable format for model training and evaluation. Exploratory data analysis is conducted to examine and understand the data to uncover patterns, find anomalies, test hypotheses, and check assumptions. Feature engineering is the process of using domain knowledge to extract features (attributes and variables) from raw data to improve the performance of the ML model. It is used to create new features or modify existing ones to make them suitable for ML algorithms. The dataset is split into training and testing subsets to build and validate the ML model, which is followed by an evaluation process with related metrics. Finally, model deployment is the implementation of the evaluated ML model in a real-world environment, which is also continuously monitored for its performance.

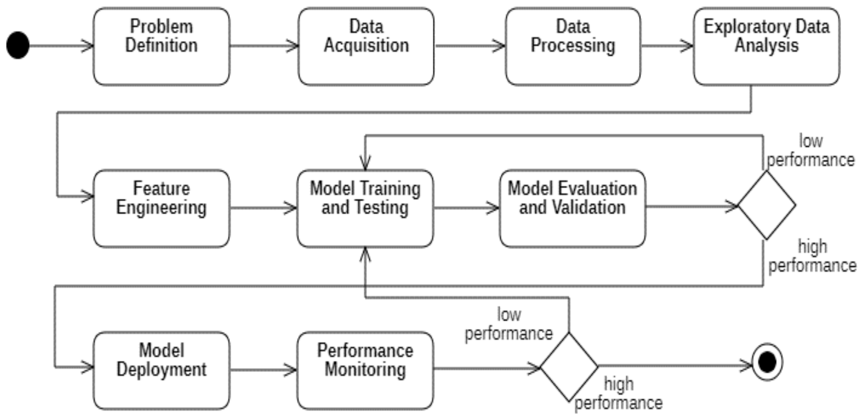

Figure 1. Machine learning process model.

3.2. Machine learning for healthcare

The healthcare industry, as one of the most critical domains, expects significant benefits from MLESP [31]. ML healthcare projects can offer transformative possibilities, such as improving patient outcomes, optimizing treatment plans, and enhancing operational efficiency and resource allocation. In this complex and safety-critical field, effective PM ensures that ML applications align with healthcare goals, comply with regulatory requirements, and address challenges related to data quality, security, and privacy. PM is essential for this rapidly evolving domain, which ensures the delivery of safe, effective, and efficient healthcare solutions. Therefore, PM remains a critical and evolving topic in healthcare practices, and research, attracting interest from practitioners and researchers across various domains [8].

Figure 2 illustrates an ML-supported illness and disease diagnosis process model. Initially, the doctor reviews the patient's clinical history after admission. Following a physical examination, the doctor may request diagnostic tests, such as blood samples, laboratory work, and radiology. Depending on the patient's condition, a consultation with other experts might be necessary. The doctor then uses the ML service to support decision-making and diagnosis. If there is any inconsistency or doubt about the ML inferences, or if the ML service is not operational, the doctor completes the diagnosis using traditional procedures.

3.3. Method engineering

ME covers a broader spectrum of development processes, activities, and guidelines, allowing for the creation or customization of methods tailored specifically for software and information system development [32]. Within ME, the decision to either construct a new method from the ground up or tailor an existing one depends on situational factors that could pertain to the organizational, process, or individual project levels. At the core of this process lies method rationale, serving as a critical background that provides the reasoning and arguments behind method prescriptions. It explains why a method user might opt to follow a method as-is or adapt it in a particular manner, thus bringing a crucial element of context awareness to ME.

In the context of ME processes, the utilization of method parts is essential with these parts manifesting as method fragments, method chunks, or method components [11]. A method fragment typically has a singular focus, concentrated either on the product or the process aspect. On the other hand, a method chunk combines a process-focused fragment with a product-focused one. The concept of method components bears a resemblance to the employment of software components within SE. A method component encompasses input and output product(s) and can function either independently or in conjunction with other method components. There are three distinctive ME approaches:

- Paradigm-Based Method Engineering (PB-ME) provides the extension and adaptation of a metamodel (i.e., SPEM 2.0) to specific situations, processes, and product models [44].
- Assembly-Based Method Engineering (AB-ME) enables the decomposition, aggregation, and tailoring of SE practices, tasks, and roles in the forms of method fragments, chunks, or components, which are stored as method parts in the method base.
- Configuration-Based Method Engineering (CB-ME) is primarily based on the concept of method components. The components are selected and then they are classified into templates to support the method configuration process.

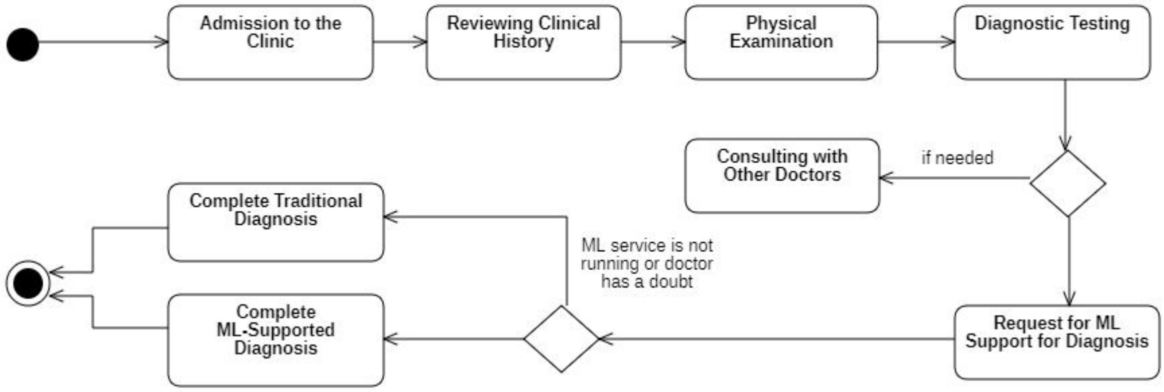

Figure 2. ML-supported illness and disease diagnosis process model method.

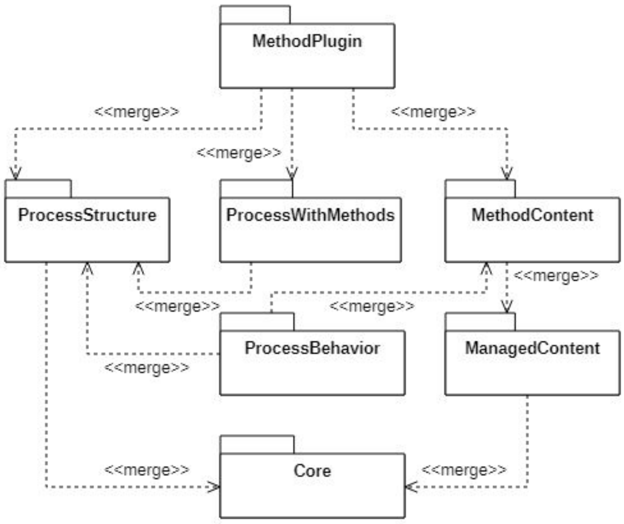

Figure 3. SPEM 2.0 Metamodel.

The concepts of the "ideal-typical method," "situational method," and "method-in-action" provide core perspectives on the knowledge and execution of software development methods [11]. The "ideal-typical method" represents a theoretical or conceptual framework of a method. The "situational method" recognizes that software development projects are not one-size-fits-all endeavors. Instead, they are influenced by various factors, including project type, complexity, domain, team composition, and organizational culture. Situational methods emphasize the need for adaptability and tailoring. The "method-in-action" represents the practical execution of the method within a real-world project. It is the embodiment of the ideal-typical method and the situational method within actual project contexts.

3.4. Eclipse Process Framework

A meta-model for ME should be able to represent any relevant method and product and process aspects of PM should be integrated within the meta-model. Software & Systems Process Engineering Metamodel (SPEM v2.0) is an Object Management Group standard for software and systems process engineering meta-model specifications

(Figure 3) [33]. As a conceptual framework, "it separates reusable method content from their application in processes". The goal is to satisfy the requirements of many processes regardless of development approaches, lifecycle models, cultural backgrounds, or levels of formalism. As it is conceptually based on SPEM 2.0, Eclipse Process Framework (EPF) is an extensible framework for software process engineering [34]. EPF Composer (EPFC) is the ME platform and method development environment [35]. The fundamental principle of EPFC is to separate the reusable method contents from their application in software development processes. Therefore, it enables designing methods and processes, authoring, tailoring, configuring, and finally, publishing these methods by providing content and library management mechanisms.

3.5. Process Algebra

PA is a mathematical structure that adheres to fundamental axioms, which enables the adoption of an algebraic and axiomatic approach for both reasoning and performing calculations for processes [36, 37]. PA is focused on equational reasoning and provides a set of formal notations, rules, and equations designed for describing algebraic manipulations of processes. Both as a formal method and mathematical framework, PA finds utility in specifying and evaluating process models, offering a range of algebraic methods to describe, specify, and verify different systems.

PA employs three fundamental approaches to describe the semantics of sequential systems: operational, denotational, and algebraic [38-40]. Each approach provides a unique perspective on modeling and understanding system behavior. The operational approach views a program as a labeled transition system, effectively capturing how a system progresses from one state to another through labeled transitions. This approach finds use in the Calculus of Communicating Systems (CCS). The denotational approach involves mapping a language to an abstract model, which helps grasp the fundamental meaning of the system's behavior. Communicating Sequential Processes (CSP) relies on this approach. Finally, the algebraic approach employs a set of algebraic rules and definitions to model the semantics of various constructs and components within the system. This approach serves as the foundation for the Algebra of Communicating Processes (ACP). Therefore, the possible contributions of PA to process design may be as follows:

- Axiomatic reasoning: PA facilitates an axiomatic approach, enabling formal reasoning and calculations with processes.
- Operational semantics: PA is equipped with operational semantics, enabling the description of system evolution through transitions.
- Verification and validation: PA provides means to control and ensure completeness, consistency, specification adherence, implementation correctness, and verification in communicating systems.
- Formal design basis: PA serves as a foundational, formal, and mathematical basis for designing and developing process models and systems.

3.5.1. Definitions and rules

Abstract algebra, as applied in PA, focuses on the study of fundamental arithmetic operations on processes. This generality is achieved through the axiomatic definition of operations. For instance, ACP employs equational axioms to abstract away from the specifics of considered processes, providing an equational framework for asynchronous process cooperation via synchronous communication. Syntax serves as a fundamental component of PA, comprising operators, a set of rules, process terms, and symbols. Collectively, these elements establish a rigorous and mathematical foundation for understanding PA semantics. Operators are used to describe sequential, parallel, and nondeterministic process compositions. For instance, in ACP, the symbol (+) signifies alternative composition, (.) represents sequential composition, and (‖) denotes parallel composition. The δ operator signifies process deadlock or failure, while the $\partial_{\mathcal{H}}$ operator encapsulates processes.

- x + y = y + x (commutativity of alternative composition) (1)
- x + (y + z) = (x + y) + z (associativity of alternative composition) (2)
- x + x = x (idempotency of alternative composition) (3)
- (x + y) . z = x . z + y . z (right distributivity of + over.) (4)
- (x . y) . z = x . (y . z) (associativity of sequential composition) (5)
- x ‖ y = y ‖ x (commutativity of parallel composition) (6)
- (x ‖ y) ‖ z = x ‖ (y ‖ z) (associativity of parallel composition) (7)
- δ operator is for deadlock or failure (8)
- ∂_H operator is for encapsulation (9)
- ⊲ (true) and ⊳ (false) are the conditional operators (10)
- √ represents successful termination of a process (11)

3.5.2. CRISP-DM

CRISP-DM method stands as one of the earliest standard process models for data mining projects, dating back to 2000 [41]. Its framework includes six sequential phases (Figure 4). While these phases are typically executed sequentially, CRISP-DM may follow a cyclic process model that allows for iterative refinement. Despite its comprehensiveness and detailed process descriptions, it does not prescribe specific PM roles. This model is relatively mature and well-established, offering a sequence of tasks and a roadmap for DS projects. However, it falls short of providing the agility and flexibility required by contemporary SE and ML projects [42].

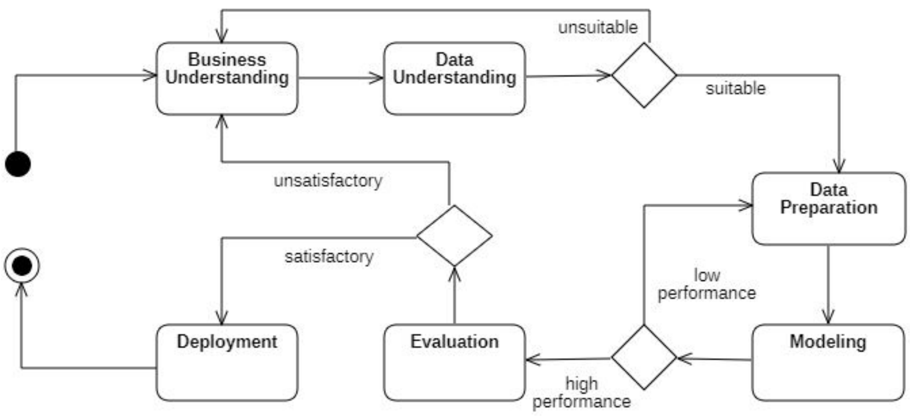

Figure 4. CRISP-DM process model.

3.5.3. Team Data Science Process

Microsoft introduced the Team Data Science Process (TDSP) as an agile and iterative PM method designed for data analytics and the development of AI applications [43]. TDSP represents a combination of SE methods, such as Scrum, and DS methods, notably CRISP-DM (Figure 5). Roles within TDSP include the project manager, project lead, solution architect, data scientist, data engineer, and application developer. It allows for the adoption of different PM approaches, whether they align more with CRISP-DM's predictiveness or the incremental and iterative nature of agile methods.

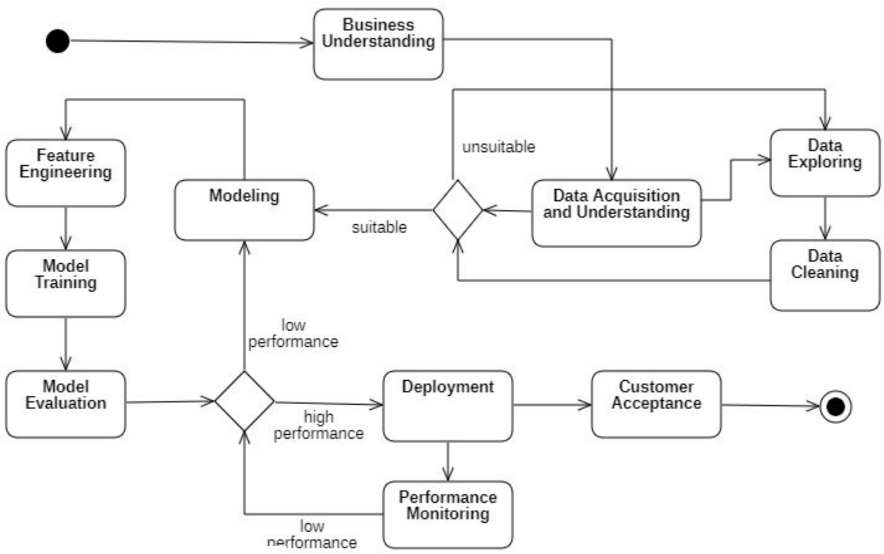

Figure 5. TDSP process model.

3.5.4. Scrum

Scrum is the most widely adopted agile PM method, characterized by its agile development approach [44]. Scrum's iterative and incremental nature, combined with its emphasis on regular communication and adaptability, makes it a popular choice for agile PM. It organizes software development into time-boxed cycles known as "sprints", lasting from two to six weeks (Figure 6). Each sprint comprises the following activities: Sprint planning meeting, sprint execution, daily standup meeting, sprint review meeting, and sprint retrospective meeting. The development team, scrum master, and product owner are the key roles within Scrum.

3.5.5. Kanban

Kanban may be regarded as one of the flexible and adaptable agile methods (Figure 7). It shares strong ties with lean thinking and just-in-time production concepts from the industry [45]. The primary objectives of Kanban are to maximize value, minimize resource waste, and prevent bottlenecks during the development cycle. It achieves these goals by effectively balancing work demands with the team's available capacity. Unlike some other agile methods, Kanban doesn't impose a predefined process model, specific team roles, and mandatory meetings. The task board serves to visualize the workflow. Kanban PM method operates based on following core principles: Limiting work-in-progress, measuring, and managing workflow, implementing quick feedback loops, and adjusting cadence

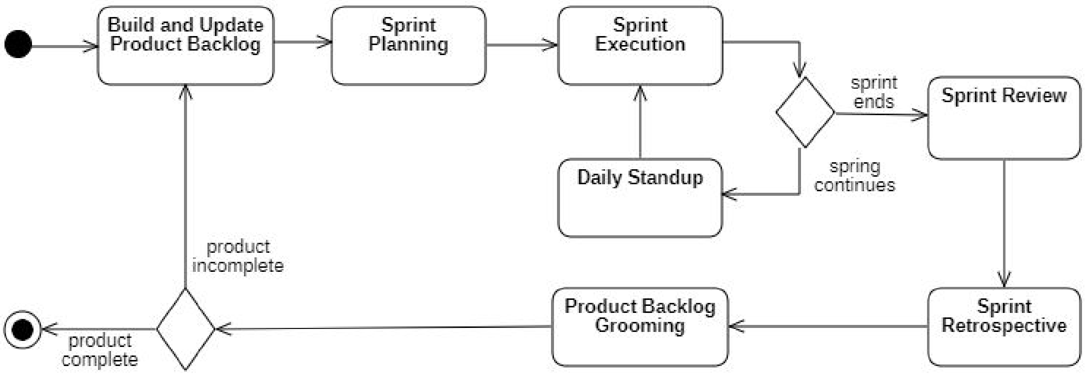

Figure 6. Scrum process model**.**

3.5.6. Data-Driven Scrum

Data-Driven Scrum (DDS) [46] is a method designed specifically for managing agile DS projects, providing an agile lean process framework tailored to the unique needs of DS. It views a DS project as a series of iterative experiments and seeks to integrate the core structure of

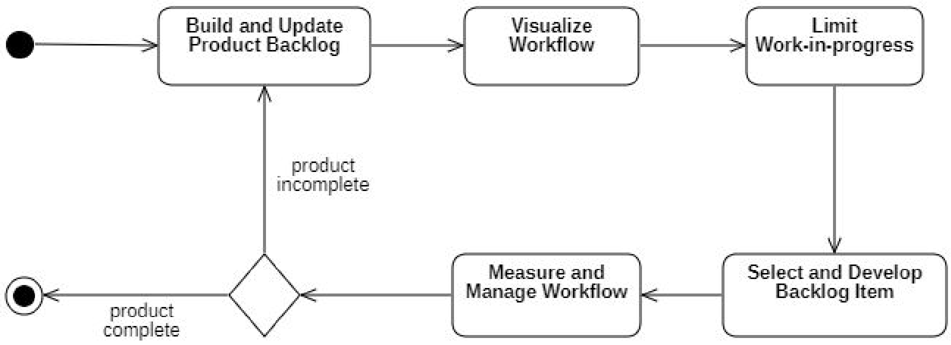

Figure 7. Principle-based Kanban process model.

Scrum with principles inspired by Kanban (Figure 8). The central objective of each iteration in DDS is to formulate, conduct, and observe a data-driven experiment, followed by a careful analysis of the results. DDS shares some

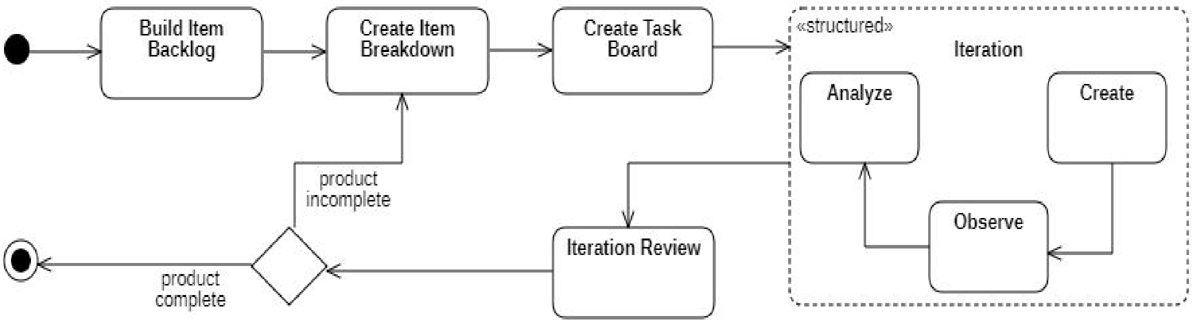

Figure 8. DDS process model.

similarities with Scrum in terms of roles, events, and the use of item backlogs. However, it introduces several distinctive elements. Decoupling meetings from the iteration, high-level item estimation, and capability-based execution are its core principles. An iteration typically encompasses three key phases: Experiment creation, performance observation, and results analysis.

3.5.7. Hybrid Methods

Relying only on a plan-driven method may lead to inflexibility, while adopting a fully agile approach may not always align with complex systems featuring interdependent components. The hybrid PM approach bridges these two extremes on a spectrum for a tailored PM strategy (Figure 9). Customizing agile PM to align with organizational needs and the specific requirements of MLESP can be highly beneficial, contingent upon the organizational context and the team's collective experiences. Various adaptations of PM methods like CRISP-DM, Scrum, Kanban, and TDSP have been successfully applied in ML projects and they can be just as effective as fully agile approaches [47].

3.5.8. Criteria for Tailoring PM Methods

Depending on the theoretical background, literature review and industrial case reports, it is possible to summarize the criteria and qualitative metrics, which can be used for tailoring PM methods according to the adopted development approaches as follows:

Table 1. Criteria and qualitative metrics for PM approaches

Criterion	Plan-Driven	Agile	Hybrid
C-1: It supports a sequential process model.	High	Low	Medium
C-2: It supports an incremental and iterative process model.	Low	High	Medium
C-3: It supports ME approaches (PB-ME, AB-ME, or CB-ME).	High	High	High
C-4: It supports the domain-specific requirements of SE.	Low	High	Medium
C-5: It supports the domain-specific requirements of ML.	High	Medium	Medium
C-6: It supports the domain-specific requirements of healthcare.	High	Medium	Medium
C-7: It has sound and satisfactory scientific evidence.	High	High	Medium
C-8: It has applications and wide acceptance in the industry.	High	High	Medium

4. Research Method

This three-phased research study employed a mixed approach that harmonizes the principles and guidelines of Design Science Research (DSR), PM, and ME (Table 2). This approach combined DSR cycles with PM and ME activities, allowing them to complement each other. Hevner et al. [48] define DSR as a research paradigm where designers seek to address problems by creating innovative artifacts, thus contributing new knowledge to the scientific body of evidence. These artifacts are intentionally constructed by humans, and they are expected to be both useful and fundamental in addressing specific problems. DSR artifacts may belong to categories, such as constructs (vocabulary and symbols), models (abstractions and representations), methods (algorithms and practices), instantiations (implemented or prototype systems), and refined design theories. This research study includes two main artifacts. The first artifact is the instantiation of a method base for tailoring PM. The second artifact is a new hybrid PM method for MLESP. PA was used to connect the processes of diverse knowledge domains. It also facilitated the formal specification, implementation, and evaluation of the PM process models and tailoring processes throughout the design and development cycles. Additionally, EPFC served as an integrated development environment for the ME processes, while providing the theoretical and implementation foundations upon which the artifacts are constructed.

4.1. Phase-1 (Relevance):

4.1.1. Step-1: Problem definition

Baskent University Hospital Ankara (BUHA) is the first and one of the most sophisticated transplantation surgery hospitals in Turkey. It uses healthcare information systems (HEIS) and owns state-of-the-art medical equipment. Recent developments in technology and ML have motivated the BUHA administration and researchers to take a step forward in the area of ML-driven HEIS. However, the stakeholders of BUHA, especially the technical staff and software developers, are not sure about "what-to-do" and "how-to-do" as well as how to adapt a PM method suitable for SE, ML, and healthcare domains [6, 8, 31]. Therefore, BUHA's goals involve a wide range of objectives, reflecting the complex nature of healthcare and the integration of ML and SE into its operations. Baskent University and its boards approved and provided support for the research. The research problem was focused on the need for a knowledge base for PM and a tailored PM method that can effectively address the domain-specific requirements.

Table 2. Research method

Phase-1 (Relevance)	Phase-2 (DSR Cycle-1)	Phase-3 (DSR Cycle-2)
Step-1: Problem definition, Step-2: Stakeholder concerns and research goals, Step-3: Evaluating available PM methods, Step-4: Goals of a new PM method as acceptance criteria Step-5: Formal specifications of available PM methods.	Step-1: Establishment of EPFC as a method base, Step-2: Creating ME content packages for SE, PM, ML, and healthcare domains.	Step-1: Identifying textual descriptions of PA specifications related to the method goals, Step-2: Transforming textual descriptions to corresponding method components, Step-3: Composition of a new phase-based PM method. Step-4: Validation of the new PM method
Output: Problem definition, available PM methods and acceptance criteria	**Output:** Artifact-1 (an ME framework and a method base)	**Output:** Artifact-2 (A new hybrid PM method for MLESP)
Theoretical Foundations and Knowledge Base: Machine Learning; PM Methods; Method Engineering; Process Algebra; Healthcare Domain		

4.1.2. Step-2: Stakeholder concerns and research goals

The ML-supported illness and disease diagnosis process, as depicted in Figure 2 above, highlights how ML can enhance decision-making in healthcare. It demonstrates that ML is used as a supportive service, with doctors having the flexibility to return to their traditional diagnosis processes when needed. A breakdown of the main concerns of the stakeholders are as follows:

- ML developers: Their focus was on the ML pipeline processes, data management, and the quality and efficiency of ML model development and deployment.
- Software engineers: They were concerned with establishing effective and efficient software engineering processes.
- IT staff: Their primary responsibility was to maintain the overall IT healthcare infrastructure, ensuring the availability and quality of healthcare systems.
- Doctors and medical staff: They were concerned with diagnostic accuracy and the quality of healthcare decision support. ML applications can help them for accurate insights and recommendations for patient care.
- Administration: Their expectations were volume-based care, profitability, and cost reduction through effective PM.

4.1.3. Step-3: Evaluating available PM methods

The selection of specific methods, including CRISP-DM, TDSP, Scrum, Kanban, and DDS, reflects the recognition that different aspects of the project may benefit from diverse PM methods. By considering a combination of these methods, a team can tailor the PM approach to specific project phases, tasks, and team dynamics. This approach can allow for a more adaptive and context-aware PM strategy, which is essential in the complex and evolving domain of ML-driven healthcare service. The ability to leverage both plan-driven and agile practices can provide a robust foundation for managing diverse aspects of the project effectively. The research goals, ML context, organizational culture, and literature review confined selecting plan-driven method (CRISP-DM), hybrid method (TDSP), and agile methods (Scrum, Kanban, and DDS) for the initial evaluation processes (Table 3):

Table 3. Evaluation of available PM methods based on the acceptance criteria

	C-1	C-2	C-4	C-5	C-6	C-7	C-8	Mean Score	Evaluation
CRISP-DM	3	2	1	3	2	3	3	2.42	Included (+)
TDSP	1	3	2	3	2	1	2	2.00	Included (+)
Scrum	1	3	3	1	2	3	3	2.28	Included (+)
Kanban	1	3	3	2	1	3	3	2.28	Included (+)
DDS	1	2	1		1	1	1	1.28	Excluded (-)
Low: 1 Medium: 2 High: 3									

The evaluation of the available PM methods was on the mean score (mean score >= 2), which was expected to be two or above according to the metrics (C-1 to C-8). Accordingly, DDS (1.28 points) is excluded from the next phases of the study. CRISP-DM (2.42 points), TDSP (2.00 points), Scrum (2.28 points), and Kanban (2.28 points) were considered for inclusion in the ME processes. It's important to note that the mean scores provided a quantitative basis for decision-making, but other factors, such as organizational requirements, team expertise

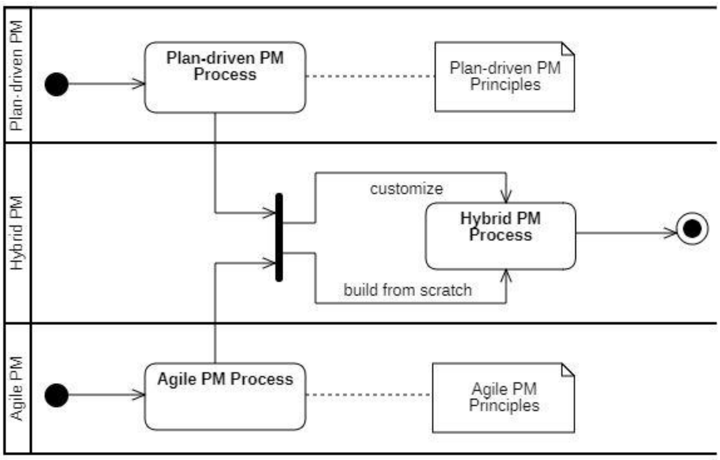

Figure 9. Hybrid process model for PM.

would be considered when finalizing the choice of PM methods.

4.1.4. Step-4: Goals of a new PM method as acceptance criteria

The method goals (MG) derived from the stakeholders' concerns reflect the main requirements of BUHA. The goals provided a holistic view of the expectations and supported the acceptance criteria for evaluating and customizing the new PM method within the context of BUHA. Each goal reflected one or more specific aspects and expectations as follows:

- MG-1 (Addressing project and organizational requirements): The PM method should accommodate both project-level and BUHA requirements, including aspects such as project type, organizational culture, and healthcare domain regulations (alignment with business goals).
- MG-2 (Early handling of ML requirements): The PM method should enable the early elicitation, analysis, and specification of data-driven, complex, and interdependent ML requirements, particularly during the RE phase (it aligns with criterion C-1).
- MG-3 (Agile and flexible approach): The PM method should allow for an agile, iterative, and incremental approach to ML and software development, with flexibility in incorporating agile and SE practices as needed during the development phase (it aligns with criterion C-2).
- MG-4 (Support for experimentation): Given the experimental nature of ML projects, especially during model training and evaluation, the PM method should enable the implementation of quick feedback loops, and work-in-progress limits based on experimentation results, akin to Kanban principles (it aligns with criteria C-2 and C-5).
- MG-5 (Visualization and predictive capabilities): The PM method should provide a structured way to visualize tasks and workflows, allowing for various types of predictions to measure and manage project tasks effectively (it aligns with criterion C-2).
- MG-6 (Domain-specific roles and communication): The PM method should define team structures and roles that align with the specific requirements of SE and ML domains, fostering communication, ownership, and knowledge sharing among team members (it aligns with criteria C-1 and C-2).
- MG-7 (Integration, monitoring, delivery, and maintenance): Recognizing the safety-critical importance of ML-driven HEIS, the PM method should support integration, monitoring, delivery, and maintenance processes to ensure the successful implementation of ML in healthcare (alignment with monitoring and maintenance processes).

4.1.5. Step-5: Formal specifications of available PM methods

In addition to the textual guidance and acceptance criteria provided by the method goals, there was a need for a mechanism both for the elimination of the researcher's subjectivity and for a common ground that would connect the knowledge domains of ML and SE during the ME processes. Therefore, PA specifications were used to meet these requirements.

4.1.5.1. Project management approaches

- Specification-1: Let a formal specification of a simple plan-driven software process model ($SPEC_{Plan-Driven}$) be as follows:

$$SPEC_{Plan-Driven} = P_{ANL} \, . \, P_{PL} \, . \, P_{DSN} \, . \, P_{DVL} \, . \, P_{TST} \quad (12)$$

where the specifications of sub-processes such as, analysis (P_{ANL}), plan (P_{PL}), design (P_{DSN}), development (P_{DVL}) and test (P_{TST}) are abstracted and sequentially composed.

- Specification-2: Let a formal specification of an agile (iterative and incremental) software process model ($SPEC_{Agile}$) be as follows:

$$SPEC_{Agile} = SPEC_{Iteration-1} \, . \, SPEC_{Iteration-2} \, . \, SPEC_{Iteration-n} \quad (13)$$

where the specifications of iterations are sequentially composed and each iteration includes the sub-processes such as, analysis (P_{ANL}), plan (P_{PL}), design (P_{DSN}), development (P_{DVL}) and test (P_{TST}).

4.1.5.2. Machine learning

Specification-3 (ML process): Let a high-level formal specification of an ML process model be as follows:

$$SPEC_{ML} = SPEC_{RQ} \, . \, SPEC_{DP} \, . \, SPEC_{MD} \, . \, SPEC_{MDPM} \quad (14)$$

where the sub-specifications such as, requirements ($SPEC_{RQ}$), data ($SPEC_{DP}$), model development ($SPEC_{MD}$), model deployment and performance monitoring ($SPEC_{MDPM}$) are abstracted and sequentially composed.

Specification-4: The data sub-process ($SPEC_{DP}$) encapsulates the inner processes: Data acquisition (P_{DA}), data processing (P_{DP}), exploratory data analysis (P_{EDA}), and feature engineering (P_{FE}) where parallel composition and encapsulation operations are applied as follows:

$$SPEC_{DP} = \partial_{\{P_{DA},\ P_{DP},\ P_{EDA},\ P_{FE}\}}(P_{DA} \parallel P_{DP} \parallel P_{EDA} \parallel P_{FE}) \quad (15)$$

Specification-5: The model development ($SPEC_{MD}$) sub-process encapsulates the inner processes: model training (P_{MTR}), model testing (P_{MTS}), and model evaluation (P_{MEV}) where sequential composition and encapsulation operations are applied as follows:

$$SPEC_{MD} = \partial_{\{P_{MTR},\ P_{DP},\ P_{MTS},\ P_{MEV}\}}(P_{MTR} \, . \, P_{MTS} \, . \, P_{MEV}) \quad (16)$$

Specification-6: The model deployment and performance monitoring sub-process ($SPEC_{MDPM}$) encapsulates the model deployment (P_{MD}) and performance monitoring (P_{PM}) inner processes where sequential composition and encapsulation operations are applied as follows:

$$SPEC_{MDPM} = \partial_{\{P_{MD},\ P_{PM},\}}(P_{MD}\ .\ P_{PM}\ ) \tag{17}$$

Specification-7: Let P_{PM} be a performance monitoring process and X_H (high level) and X_L (low level) are the constants that represent the performance states (X_n) of an ML model. The specifications for a continuous and recursive P_{PM} are as follows:

$X_H = P_{PM}\ X_L$ and $X_L = P_{PM}\ X_H$, are the two equations that represent the performance states of a P_{PM}, which would be at a high or low level. (18)

$SPEC_{MDPM} \lhd X_n \rhd SPEC_{MD}$ which also means that if the state of P_{PM} is X_L then the model development processes of $SPEC_{MD}$ are revisited, otherwise the processes of $SPEC_{MDPM}$ are executed. (19)

4.1.5.3. CRISP-DM

Specification-8: Let a high-level formal specification of a CRISP-DM process model be as follows:

$$SPEC_{CRISP} = SPEC_{BU}\ .\ SPEC_{DU}\ .\ SPEC_{MDDP} \tag{20}$$

where the sub-specifications such as, business understanding ($SPEC_{BU}$), data understanding ($SPEC_{DU}$), model development and model deployment ($SPEC_{MDDP}$) are abstracted and sequentially composed.

Specification-9: The model development and model deployment specification ($SPEC_{MDDP}$) encapsulates the data preparation (P_{DP}), modeling (P_M), evaluation (P_E) and model deployment (P_{DPL}) inner processes where sequential composition and encapsulation operations are applied as follows:

$$SPEC_{MDDP} = \partial_{\{P_{DP},\ P_{M},\ P_{E}, P_{DPL}\ \}}(P_{DP}\ .\ P_M\ .\ P_E\ .P_{DPL}\ )$$
..(21)

Specification-10: Let P_E be an evaluation process, and X_H (high level) and X_L (low level) be the constants that represent the performance states (X_n) of an ML model of $SPEC_{CRISP}$. The specifications for a continuous and recursive P_{DPL} are as follows:

$X_H = P_{DPL}\ X_L$ and $X_L = P_{DPL}\ X_H$, are the two equations that represent the performance states of a P_{DPL}, which would be at a high or low level. (22)

$SPEC_{BU} \lhd X_n \rhd P_{DPL}$; which also means that if the state of P_{DPL} is X_L then the business understanding processes of $SPEC_{BU}$ are revisited, otherwise the processes P_{DPL} is conducted. (23)

4.1.5.4. TDSP

Specification-11: Let a high-level formal specification of a TDSP process model be as follows:

$$SPEC_{TDSP} = SPEC_{BDU}\ .\ SPEC_{MMDP}\ .\ SPEC_{CA} \tag{24}$$

where the sub-specifications such as, business and data understanding ($SPEC_{BDU}$), modeling and model deployment ($SPEC_{MMDP}$), and customer acceptance ($SPEC_{CA}$) are abstracted and sequentially composed.

Specification-12: The modeling and model deployment sub-specification ($SPEC_{MDDP}$) encapsulates the feature engineering (P_{FE}), modeling training (P_{MT}), model evaluation (P_{ME}), model deployment (P_{DPL}), and performance monitoring (P_{PM}) inner processes where sequential composition and encapsulation operations are applied as follows:

$$SPEC_{MMDP} = \partial_{\{P_{FE},\ P_{MT}, P_{ME}, P_{DPL}, P_{PM}\}}(P_{FE}\ .\ P_{MT}\ .\ P_{ME}\ .\ P_{DPL}\ .\ P_{PM}) \tag{25}$$

Specification-13: Let (P_{PM}) be a performance monitoring process, and X_H (high level) and X_L (low level) are the constants that represent the performance states (X_n) of an ML model of $SPEC_{TDSP}$. The specifications for a continuous and recursive (P_{PM}) are as follows:

$X_H = P_{PM}\ X_L$ and $X_L = P_{PM}\ X_H$, are the two equations that represent the performance states of a P_{DPL}, which would be at a high or low level. (26)

$SPEC_{MMDP} \lhd X_n \rhd\ P_{PM}$; which also means that if the state of P_{PM} is X_L then the modeling and model deployment processes ($SPEC_{MMDP}$) are revisited, otherwise the processes P_{PM} is conducted (27)

4.1.5.5. Scrum

Specification-14: A sprint execution process (P_{SE}) conducts the software processes such as analysis (P_{ANL}), plan (P_{PL}), design (P_{DSN}), development (P_{DVL}) and test (P_{TST}) sequentially as follows:.

$$P_{SE} = P_{ANL}\ .\ P_{PL}\ .\ P_{DSN}\ .\ P_{DVL}\ .\ P_{TST} \tag{28}$$

Specification-15: The Scrum process model ($SPEC_{Scrum}$) executes the sprint processes iteratively and incrementally as follows:

$$SPEC_{Scrum} = \partial_{\{P_{Sprnt-1},\ P_{Sprint-2},\dots,\dots, P_{Sprint-n}\ \}}\ P_{Sprint-1}\ .\ P_{Sprint-2}, \dots,\ P_{Sprint-n} \tag{29}$$

where the sequential and encapsulation operations are applied to the sprint processes.

Specification-16: Let the Scrum process model ($SPEC_{Scrum}$) be as follows:

$$SPEC_{Scrum} = \partial_{\{P_{SP},\ P_{SE},\ P_{SRV}, P_{SRT}, P_{PG}\ \}}\ P_{SP}\ .\ P_{SE}\ .\ P_{SRV}\ .\ P_{SRT}\ .\ P_{PG} \tag{30}$$

where the sequential and encapsulation operations are applied to the Scrum processes such as, sprint planning (P_{SP}), sprint execution (P_{SE}), sprint review (P_{SRV}), sprint retrospective (P_{SRT}) and product backlog grooming (P_{PG}).

Specification-17: Let X_{PB} be a constant that represents the number of backlog items in a product backlog:

$P_{SP} \lhd X_{PB} \rhd \surd$, which means that if $X_{PB} > 0$ then the sprint planning process (P_{SP}) is revisited otherwise the Scrum process terminates successfully. (31)

4.1.5.6. Kanban

Specification-18: A Kanban software process (P_{KS}) includes the software processes such as analysis (P_{ANL}), design (P_{DSN}), development (P_{DVL}) and test (P_{TST}) iteratively and incrementally:

$$P_{KS} = P_{ANL} \cdot P_{DSN} \cdot P_{DVL} \cdot P_{TST} \tag{32}$$

Specification-19: Let the Kanban process model ($SPEC_{Kanban}$) be as follows:

$$SPEC_{Kanban} = \partial_{\{P_{PB},\ P_{VW},\ P_{LW},P_{SD},P_{MM}\}} P_{PB} \cdot P_{VW} \cdot P_{LW} \cdot P_{SD} \cdot P_{MM} \tag{33}$$

where the sequential and encapsulation operations are applied to the Kanban processes such as, build and update product backlog (P_{PB}), visualize workflow (P_{VW}), limit work-in-process (P_{LW}), select and develop backlog item (P_{SD}), measure and manage workflow (P_{MM}).

Specification-20: Let X_{PB} be a constant that represents the number of items in a product backlog:

$P_{PB} \lhd X_{PB} \rhd \surd$; which means that if $X_{PB} > 0$ then the product backlog process (P_{PB}) is revisited otherwise Kanban process terminates successfully. (34)

4.1.5.7. Hybrid methods

Specification-21: Let a high-level formal specification of a hybrid process model be as follows:

$$SPEC_{Hybrid} = SPEC_{Plan-Driven} \parallel SPEC_{Agile} \tag{35}$$

where the specifications for plan-driven process model ($SPEC_{Plan-Driven}$) and agile process model ($SPEC_{Agile}$) are abstracted and parallel composition is applied.

4.2. Phase-2 (DSR Cycle-1)

At this phase, the focus was on creating ME content packages related to ML, SE, PM methods, and the healthcare domain of BUHA. The primary output of this phase is the development of a method base and an ME framework. The method parts of selected PM methods were created using EPFC. These parts were designed to align with the method goals and the characteristic requirements of the MLESP.

4.2.1. Step-1 (Establishment of EPFC as a method base):

The sub-steps for establishing EPFC and the method base include:

- Defining the specific practices, processes, patterns, tasks, roles, work products, and guiding elements related to ML,
- Identifying and specifying the practices associated with ML, CRISP-DM, TDSP, Scrum, and Kanban,
- Defining the SE practices, processes, patterns, tasks, roles, work products, and guiding elements relevant to the SE domain,
- Identifying and specifying the healthcare and diagnostic practices, processes, patterns, tasks, roles, work products, and guiding elements specific to the healthcare domain,
- Determining how the method will be decomposed into smaller parts, such as fragments, chunks, or components. This step involved deciding the granularity level at which method parts would be defined,
- Creating the actual method parts (fragments, chunks, or components) within the method base according to the specifications provided in the previous steps.

4.2.2. Step-2 (Creating ME content packages for SE, ML, PM, and healthcare):

The ME content packages, and corresponding artifacts (screenshots and tables) are given in the appendices section of the paper, which are represented by figures and tables (See appendices).

4.3. Phase-3 (DSR Cycle-2)

4.3.1. Step-1 (Identification of PA specifications related to the method goals):

It is important to remind that at Phase-1, PA specifications and the PM method goals provided not only the formal and rigorous foundation but also the acceptance and validation criteria for the tailored PM method. Table 4 presents the PA specifications (goals, related method, equations and terms) that are directly related to the method goals, which are also specified in the previous sections. Note that the textual descriptions of the method parts and components are retrieved from the PA specifications of available PM methods.

4.3.2. Step-2 (Transforming textual descriptions to corresponding method components)

The textual descriptions retrieved from the PA specifications PM method are transformed to the corresponding ME components (principles, roles, tasks and products) of agile and plan-driven PM approaches as shown in Table 5.

The PA specifications in Table 4 and the corresponding ME components given in Table 5 led to the design and implementing of a hybrid PM method. As indicated below, the PA implementations of the plan-driven process model ($IMP_{Plan-Driven}$) and agile process model (IMP_{Agile}) are abstracted and the parallel composition operation is applied as follows:

$$IMP_{Hybrid\ Process\ Model} = IMP_{Plan-Driven} \parallel IMP_{Agile} \quad (30)$$

which means that the plan-driven requirements approach, and the customized agile practices are integrated to form a hybrid process model for ML PM. Therefore, the requirements process of ML (RQ), the data processes of CRISP-DM (Business understanding (BU), data acquisition and understanding (DU)), and TDSP (Business and data understanding (BDU), and feature engineering (FE)) allow implementing the Phase-1 as a plan-driven phase as formulated below:

$$IMP_{Phase-1\ (Requirements\ (Plan-driven))} = \partial_{\{P_{RQ},\ P_{BU},\ P_{DU},P_{BDU}.P_{FE}\}}(P_{RQ}\ .\ P_{BU}\ .\ P_{DU}\ .\ P_{BDU}\ .\ P_{FE}) \quad (37)$$

Table 4. Identification of PA specifications related to the method goals.

Method goal	Related PM method	Equation number	Related PA specifications of available PM methods	Textual descriptions retrieved from the PA specifications
MG-2	ML	(14)	$SPEC_{ML} = SPEC_{RQ}\ .\ SPEC_{DP}\ .\ SPEC_{MD}\ .\ SPEC_{MDPM}$	Requirements (RQ), data (DP), model development (MD), model deployment, and performance monitoring (MDPM)
	ML	(15)	$SPEC_{DP} = \partial_{\{P_{DA},\ P_{DP},\ P_{EDA},\ P_{FE}\}}(P_{DA} \parallel P_{DP} \parallel P_{EDA} \parallel P_{FE})$	Data acquisition (PDA), data processing (PDP), exploratory data analysis (PEDA), and feature engineering (PFE)
	Plan-Driven	(12)	$SPEC_{Plan-Driven} = P_{ANL}\ .\ P_{PL}\ .\ P_{DSN} \dots$	Analysis (ANL), planning (PL), design (DSN)
	CRISP-DM	(20)	$SPEC_{CRISP} = SPEC_{BU}\ .\ SPEC_{DU} \dots$	Business understanding (BU), data acquisition and understanding (DU)
	TDSP	(24)	$SPEC_{TDSP} = SPEC_{BDU} \dots$	Business and data understanding (BDU)
MG-3	Agile	(13)	$SPEC_{Agile} = SPEC_{Iteration-1}\ .\ SPEC_{Iteration-n}$	Iterative and incremental development (Iteration-1/n)
MG-3/6	Scrum	(29)	$SPEC_{Scrum} = P_{Sprint-1} \dots P_{Sprint-n}$	Iterative and incremental development (Sprint- 1/n)
MG-4/7	Kanban	(33)	$SPEC_{Kanban} = P_{PB}\ .\ P_{VW}\ .\ P_{LW}\ .\ P_{SD}\ .\ P_{MM}$	Quick feedback loop and work-in-progress limit (Kanban principles)
MG-2/3/4	Hybrid	(35)	$SPEC_{Hybrid} = SPEC_{Plan-driven} \parallel SPEC_{Agile}$	Combination of plan-driven and agile process models

Table 5. Agile and plan-driven method components (principles, roles, and products) pertaining to the goals of the new PM method.

CRISP-DM (Plan-Driven)	TDSP (Plan-Driven)	Scrum (Agile)	Kanban (Agile)
Principles			
-	-	Scrum principles	Kanban principles
Roles			
-	Project Manager, Data Scientist, Data Engineer, Application Developer	Product Owner, Scrum Team	
Tasks			
Business understanding, data preparation	Data acquisition and understanding, feature engineering	Sprint activities	
Products			
Data reports (description, exploration, quality, cleaning), deployment, monitoring and maintenance plans, final report	Charter document, data dictionary	Product backlog, release plan, sprint burndown and release burndown charts	Kanban task board, visualized workflow

where the sequential composition and encapsulation operations are applied to the processes of Phase-1 such as, P_{RQ}, P_{BU}, P_{DU}, P_{BDU} and P_{FE} .

$$IMP_{Phase-2\,(Development\,(Agile))} = \partial_{\{IMP_{Scrum},\,(IMP_{CRISP-DM} \parallel IMP_{SE})\}}(IMP_{Scrum} \cdot (IMP_{CRISP-DM} \parallel IMP_{SE})) \quad (38)$$

where the sequential and parallel composition and encapsulation operations are applied to the processes of Phase-2. Scrum encapsulates the ML modeling processes (model training, model testing, and model evaluation), the data processes of CRISP-DM and the SE practices (software analysis, software design, software development, and software testing).

$$IMP_{Phase-3\,(Deployment\,and\,Delivery)} = \partial_{\{P_{DDP},\,P_{PM}\}}(P_{DPL} \cdot P_{PM}) \quad (39)$$

where P_{DPL} is a deployment and delivery process and P_{PM} is a performance monitoring process. X_H (high level) and X_L (low level) are the constants that represent the performance states (X_n) of a deployed ML model.

$X_H = P_{PM}\ X_L$ and $X_L = P_{PM}\ X_H$, are the two equations that represent the performance states of a P_{PM}, which would be at a high or low level. $IMP_{Phase-2} \lhd X_n \rhd IMP_{Phase-3}$, which means that if the state of P_{PM} is X_L then $IMP_{Phase-2}$ is revisited, otherwise $IMP_{Phase-3}$ is executed (40)

$$IMP_{Phase-4\,(Deployment\,and\,Delivery)} = \partial_{\{P_{SMO} \parallel P_{SMA}\}}(P_{SMO} \parallel P_{SMA}) \quad (41)$$

which means that Phase-4 encapsulates the software monitoring process (P_{SMO}) and software maintenance process (P_{SMA}) where the parallel composition operation is applied. It is possible to state that the method goals, PA specifications, the BUHA stakeholder concerns and the domain-specific requirements (PM, ML, SE, healthcare) can rationalize the adoption of a hybrid PM approach. Table 6 presents the textual definitions of the phases and processes of the proposed hybrid PM method.

4.3.3. Step-3 (Composition of a New Phased-based PM Method):

The agile and plan-driven method components in Table 5 are composed to form a phase-based PM process model as given in Table 6.

As can be seen from Table 6 and Figure 10, the hybrid PM method consists of four phases: Phase-1 (Requirements Engineering Phase (REP)), Phase-2 (Development Phase (DP)), Phase-3 (Deployment and Delivery Phase (DDP)), and Phase-4 (Continuous Monitoring and Maintenance Phase (CMMP)).

The REP phase is plan-driven, the DP and DDP phases are agile, and the MMP phase is continuous. Some of the tasks of CRISP-DM and Scrum are customized for the ML processes. For example, the activities and tasks of a Scrum sprint are not time-boxed. Thus, the model training and evaluation tasks of CRISP-DM are iterated in the sprints according to the results of the ML experimentation processes (model training, test, and evaluation). Both agile and Kanban principles guide the project life cycle. The method base and the screen shot of the hybrid PM method for MLESP is given in Appendices (Figure A.6).

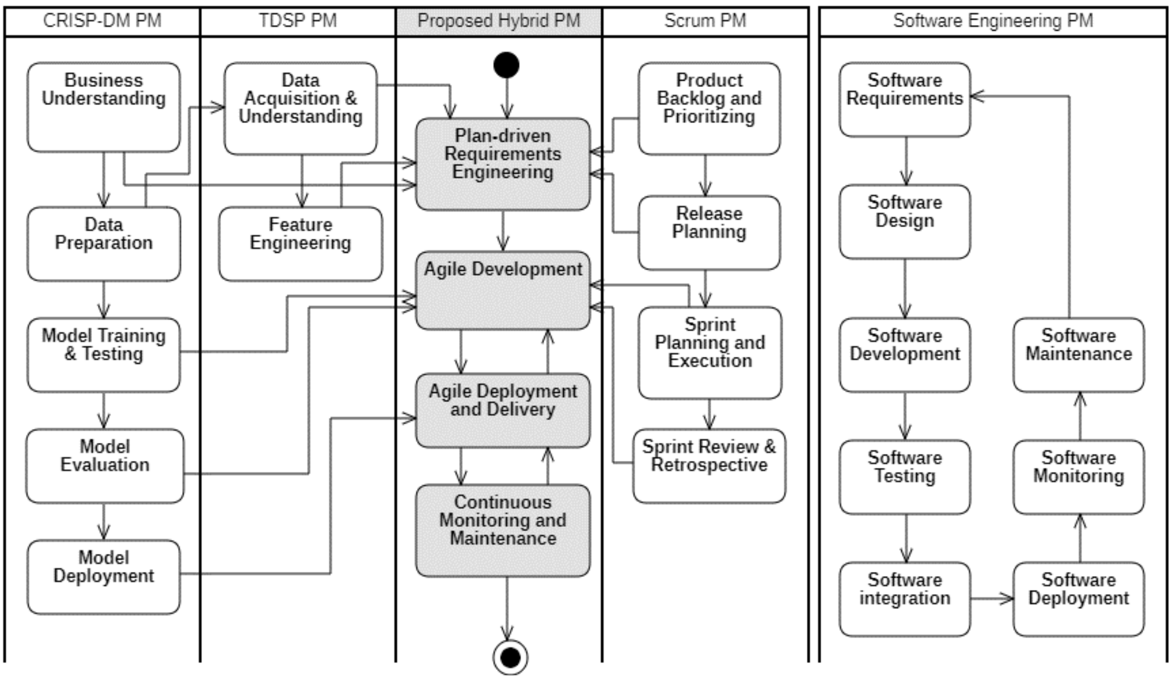

Figure 10. The process model for the new PM method.

Table 6. Phases of the proposed hybrid PM method.

CRISP-DM (Customized)	TDSP (Customized)	Scrum (Customized)	Software Engineering
Phase-1: Requirements Engineering (REP) (Plan-Driven)			
Business understanding	Data acquisition & understanding	Product backlog and prioritizing	Software requirements
Data preparation	Feature engineering	Release planning	
Phase-2: Development (DP) (Agile)			
Model training		Sprint planning	Software analysis
Model testing		Sprint execution	Software design
Model evaluation		Sprint review and sprint retrospective	Software development
			Software testing
Phase-3: Deployment and Delivery (DDP) (Agile)			
Model deployment	Deployment		Software integration
			Software deployment
Phase-4: Monitoring and Maintenance (MMP) (Continuous)			
			Software monitoring
			Software maintenance

4.3.4. Step-4 (Validation of the New PM Method):

The new hybrid PM method is both an "ideal-typical method" and a "situational method". As such, its process model provides a foundational understanding of how the development of an MLESP should ideally proceed. Therefore, the scope and limitations confined us validate the requirements processes (phase 1) of the new PM method. The use case-based scenario analysis technique was employed for checking the consistencies with the requirements and the ME components as the PM tasks, roles and responsibilities. The specified requirements were also evaluated for correctness and completeness.

4.3.4.1. Scenario

The main stakeholders and users are the administration, IT staff, medical staff (doctors, physicians, nurses, medical assistants, etc.), ML engineers, and patients. BUHA administration, with its financial viewpoint, aims to improve volume-based care and profitability by increasing the number of patients. From a medical viewpoint, doctors and medical assistants focus on enhancing diagnosis processes and accuracy. By holding a technical viewpoint, IT managers and staff prioritize the quality and availability of medical and IT operations, including the HEIS. While ML engineers share some technical concerns with IT staff, their primary focus is on ML processes, i.e. data quality and availability, data processing, and modeling processes. Finally, the patients are concerned with diagnostic accuracy and the quality of healthcare services.

4.3.4.2. Use Case

Although used interchangeably, use cases and use case diagrams are different. Use case diagrams provide a high-level, visual overview of requirements. On the other hand, use cases are narrative descriptions documenting user-system interactions and system requirements from an external perspective. They are relatively easy to write, read, and understand compared to other tools. A use case is defined as a series of interactions between external users/actors and the system in question, simply detailing “who (user/actor) does what (interaction) with the system, for what purpose (goal)”. The use case descriptions to validate ML-enabled illness and disease diagnosis are given in Table 7.

Figure 11 presents the use case diagram for illness and disease diagnosis. The “illness and disease diagnosis” boundary isolates external actors from internal use cases. The doctor, patient, and medical assistant are concrete actors, while staff is an abstract actor enabling inheritance. The doctor is a direct actor who uses the system and requests the ML service, while the patient is an indirect actor providing necessary diagnostic information and a primary actor in the “patient admission” use case. Medical staff, the appointment office, and the ML service are secondary actors who provide assistance. The “performing diagnosis” use case includes “physical examination,” “reviewing clinical history,” “diagnostic testing,” and “consulting with other clinicians.” The “machine learning process” use case extends the “performing diagnosis” use case as a special form of it.

Table 7. The use case to validate ML-enabled illness and disease diagnosis.

Identification: UC-1: Illness and disease diagnosis by using an ML service.
Goal: To improve the diagnosis process and its accuracy through an ML service.
Description: The purpose of this use case is to describe the actors, functionalities, scenarios, elements, and conditions that are required for ML-driven illness and disease diagnosis.
Actors: Medical staff (abstract), doctor (concrete, primary), physician (concrete, primary), medical assistant (concrete, secondary), patient (concrete, secondary), appointment office (concrete, secondary), ML service (concrete, secondary).
Functionalities: The functionalities for illness and disease diagnosis when using ML service.
Main scenario: Step-1: Patient admission, Step-2: Reviewing clinical history, Step-3: Physical examination, Step-4: Diagnostic testing, Step-5: Consulting with other clinicians, if necessary, Step-6: Requesting ML service for illness and disease diagnosis, Step-7: Completing illness and disease diagnosis.
Pre-conditions: The patient is admitted; the ML service is running.
Post-conditions: The illness and disease diagnosis processes are completed.
Alternative scenarios: Conducting traditional diagnostic processes.
Exceptions: Exception-1: The ML service is not working. Handling-1: Conduct the traditional diagnostic processes. Exception-2: The doctor's and the ML service's diagnoses do not correlate. Handling-2: (a) Consult with other clinicians if necessary, and (b) perform individual decision-making.
Metrics: The accuracy level of the ML service.
Relationships: It has relationships with the use cases for manual or automated ML processes.
Additional knowledge: An application programming interface (API) is needed for integrating the ML application into the current HEIS.

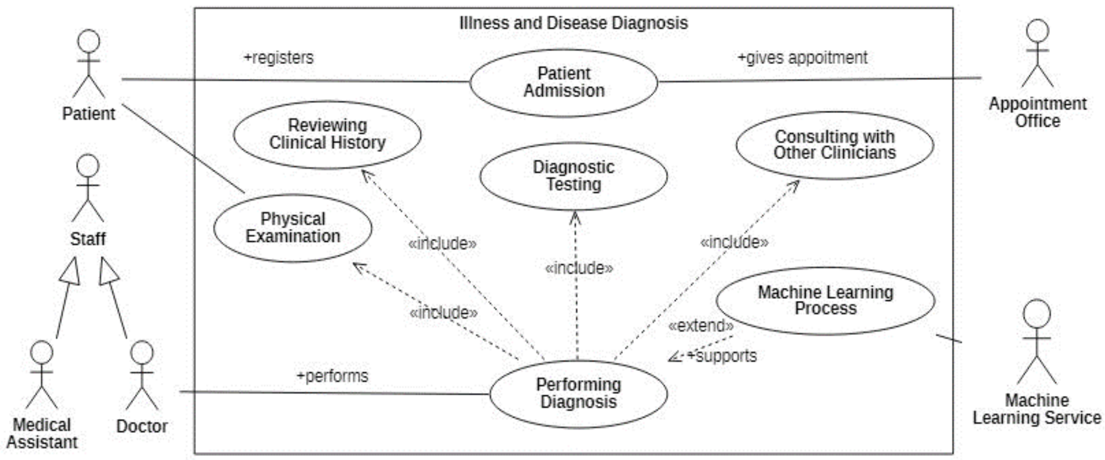

Figure 11. The use case diagram for illness and disease diagnosis.

Figure 12 presents the use case diagram for the ML process. The ML engineer is a direct, concrete, and primary actor in the "manual ML process" use case but serves as a secondary actor when monitoring the "automated ML pipeline process." Both the "manual ML process" and "automated ML pipeline process" use cases extend the main "ML process" use case. This use case includes the "data extraction," "data analysis," "data preparation," "model training," "model evaluation," "model validation," and "model deployment" use cases.

However, this approach can come with its own set of challenges, including knowledge management training [49], and adapting PM methods to suit specific contextual factors. A review of the literature reveals two primary approaches to method tailoring: The Contingency Factor Approach (CFA) and the Method Engineering Approach (MEA) [13, 50]. The CFA operates on the assumption that tailoring is necessary regardless of the selected PM method. It argues that various context features or factors (project team, internal and external environments, prior knowledge, etc.) play a pivotal role in determining the tailoring process. However, the CFA faces key challenges: (a) The team bears the primary responsibility for the tailoring process; (b) tailoring often occurs in an ad hoc manner; and (c) there is a lack of standardization and integrated platforms for tailoring. As a result, the MEA has emerged as the preferred approach in the majority of research studies, getting 69.7% preference, offering a structured and systematic approach to method tailoring, and thus, addressing the shortcomings associated with the CFA [50].

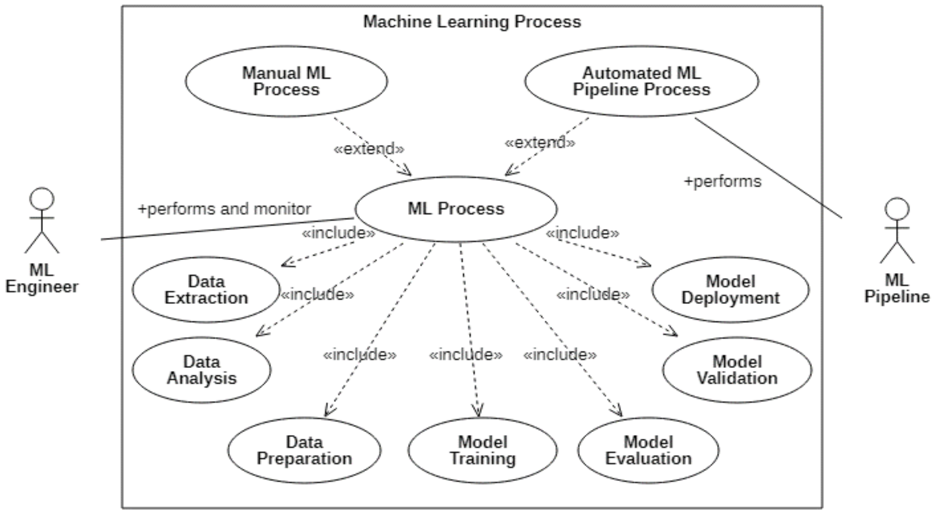

Figure 12. The use case diagram ML process.

5. Discussion

5.1. Why adopting a method base and ME approach for tailoring PM methods?

It's widely recognized that the actual implementation of a method in real projects often deviates from the ideal process models outlined in manuals [11]. The contemporary PM environments, offering both cloud and self-managed deployment options, focus on version control, software development, task management, and issue tracking. In this study, the proposed ME framework for ML can offer an intellectual knowledge base that encompasses a wide range of contents, including method practices, templates, descriptions, guidelines, and documents [5]. This knowledge base serves as the foundation for knowledge management, reference, and training. Additionally, it functions as a web-based content management system, enabling knowledge sharing to enhance PM processes. As previously discussed, team members may also encounter challenges in workflow management during a project [12, 21]. Therefore, the proposed framework addresses this by allowing the presentation of ME elements in various display formats. Furthermore, The ME framework facilitates the organization of PM processes not only in the workflow format but also in a work breakdown structure (Appendix 2).

Developing an in-house method for PM can require a sense of ownership within an organization and teams.

Establishment of a shared understanding among team members is essential. In the domain of software development, it's common practice for developers to require access to various sources or versions of the same project product and information. However, software development organizations and teams often lack formal mechanisms for in-house training and educating themselves on development PM processes and standards. Instead, they often tend to rely on ad hoc approaches to PM, formed by their past experiences and tacit knowledge. Functioning as an ME framework, the proposed PM framework can provide guidance, reusable templates, process patterns required for designing and developing tailored PM models, specifically for ML projects. The incorporation of the SPEM 2.0 meta-model specification within EPFC offers a critical advantage, which enables the separation of method content from development processes, while facilitating the formal descriptions of ME elements [33-35]. Consequently, these capabilities can contribute to the design and development of customized PM lifecycle models that can be ideally suited for ML projects.

By combining the strengths of both ME, agile, and plan-driven PM approaches, the proposed ME framework aims to provide a toolkit that enables informed decision-making on project outcomes. It is evident that as the landscape of PM continues to evolve, a flexible and hybrid approach, such as the one proposed in this paper, may be well-suited to meet the challenges of today's dynamic and diverse ML PM environments. While this study addresses these challenges and requirements of ML, SE, and healthcare domains, it would also align with the findings of other studies that highlight the plan-driven, agile, and exploratory nature of ML projects. Therefore, the method base and the ME framework can bridge the gap between these domains and contribute to the practical adoption and implementation of various PM approaches.

5.2. Why a hybrid PM method?

This study also highlights the evolving nature of software development methods and the need for a holistic approach that considers a range of factors as well as methods and practices for ML-driven software applications. The arguments related to the adoption of a hybrid PM approach align with those of other studies [30], such as reinforcing the importance of considering the plan-driven, iterative, and exploratory nature of ML projects. Traditional methods, such as the plan-driven (waterfall) stress structure and predictability. In contrast, agile approaches focus on adaptability and flexibility. Agile PM methods, which gained popularity for their adaptability and responsiveness, may not be a one-size-fits-all solution, particularly requiring a significant level of uncertainty and regulatory constraints as for ML and healthcare domain. Recognizing the limitations and strengths of each approach, the proposed hybrid PM method can emerge as an alternative PM method. It can ensure compliance with regulatory frameworks, which is critical for healthcare and finance. It also integrates agile risk management approaches with traditional risk assessment and mitigation strategies and thus enhances the project's ability to identify and respond to risks effectively.

One of our observations is that neither PM methods nor SE practices may explicitly assure or impede agility. SE practices can have a substantial impact on the degree of agility [47]. This may reinforce the idea that agile principles may be rooted in SE practices, such as continuous integration, automated testing, and iterative and incremental development. This aligns with our argument for combining different methods to achieve agility while recognizing that no single method or practice can guarantee agility and project success.

It is argued that some known PM models like CRISP-DM and TDSP can face challenges in meeting the domain-specific requirements of SE, ML, and healthcare simultaneously. Therefore, we emphasize the importance of tailoring PM approaches to specific contexts and requirements. This also aligns with the broader industry trend of adapting methodologies to suit the unique characteristics of projects. Tailoring process models is a pragmatic response to the changing demands of these domains. Our argument is also verified by Haakman et al.'s [51] case study, which identified gaps in current lifecycle models when developing AI-based systems.

Ramasamy et al.'s [52] study also underlines the iterative and exploratory nature of ML projects, a fundamental characteristic shared by many ML projects. The hybrid PM method, with its agile and iterative characteristics, can be well-suited to managing projects with such characteristics. Additionally, our proposed ME framework can enable the seamless integration of ML workflow management and environments into SE processes. This concept aligns with Lwakatare et al.'s [16] suggestions, which emphasize the need for effectively integrating ML workflows into SE environments [53]. As a result, the proposed PM method in this study can address this integration challenge, contributing to the efficiency and effectiveness of workflow management in ML projects within SE contexts.

The argument for a hybrid PM also finds strong support in different domains and applications. For example, [54] underscores the effectiveness of combining base and meta-learners in hybrid systems, demonstrating the advantages of integrating diverse methods for robust and reliable outcomes. Similarly, [55] proposes a predictive ML framework for healthcare, pointing out the importance of structured PM approaches to ensure the successful implementation and deployment of models in critical sectors. [56] further underscores the complexities of managing imbalanced datasets in ML for fraud detection, indicating the necessity of adaptable strategies that can address technical and operational challenges. From a software engineering perspective, [57] argues for tailored methods to ensure the successful execution of ML projects. [58] emphasizes the growing role of ML in healthcare and the need for frameworks that align technological solutions with sector-specific standards. [59] explores ML-based diagnostic systems, underlining the importance of structured approaches to manage technical, ethical, and operational dimensions. [60] adopts a formal approach and refers to the importance of algebra. Additionally, the Internet of Things (IoT) is one of the application domains where ML PM requires the adoption of hybrid PM methods, as the convergence of IoT and ML introduces complex data flows, real-time processing demands, and integration challenges [61, 62]. Collectively, these studies reinforce the argument that adopting a hybrid PM approach can provide the necessary flexibility and structure to manage the dynamic and complex nature of ML projects across various domains.

6. Validity and Reliability

In terms of construct validity, we based our design research on foundational theories and models of ME, PM, ML, PA, and SE. By aligning research constructs with existing literature, theory-guided interventions ensured that our designs and developed artifacts were theoretically sound. We followed a well-defined structured and iterative process in each DSR cycle, including the identification of issues and the application of ME techniques. This structured approach was to minimize the risk of confounding effects of design actions for the artifacts. In terms of internal validity, the participation, intervention, and interpretation of the researcher and stakeholders may lead to subjectivity, which may be regarded as one of the criticized attributes of DSR. Therefore, PA was employed to decrease this subjectivity and increase the internal validity. The active involvement of the researcher in DSR cycle helped validate and ensure that the changes to the artifacts were attributed to the interventions made. However, the researcher played a dominant participant role and guided the study and there were potential biases. Therefore, the employment of formal methods, visual and textual representation techniques helped address these internal validity concerns.

7. Limitations

As discussed previously, the hybrid PM method proposed in this study is both an "ideal-typical method" and a "situational method". It is worth underlining the fact that real-world projects require the adaptation of the ideal method to suit the specific situation. However, conducting empirical research to assess the effectiveness and adaptability of a PM method (method-in-action) is a resource-intensive endeavor that can span extended periods. While this study successfully laid the conceptual and rigorous foundations for the method base, the ME framework and the hybrid PM method, yet there is a need for practical validation through real-world case studies, especially in contexts that converge ML, SE, and other domains. The effort, informed by longitudinal and well-designed case studies and industry collaboration, will enhance our understanding of how our framework and hybrid PM method can perform and adapt in dynamic and multifaceted project environments.

The method-in-action limitation can be addressed by adding Statistical Thinking (ST) and quality-related tools into the mixed approach for ML PM [63]. ST's focus on understanding variation and process improvement can align well with managing complex ML processes. This addition can also enable teams to detect, assess, and reduce inconsistencies in each step of an ML project. Thus, this integration would promote proactive adjustments and risk mitigation, enhancing reliability and consistency in ML and SE projects, especially in mission-critical domains like healthcare [64].

8. Conclusions and Future Directions

In this paper, the outputs of a research study are presented that try to bridge the critical gaps in PM when dealing with the complicated relationships of ML, SE, and healthcare domains. We aim to provide a comprehensive solution that could harmonize the unique requirements of each domain while accommodating the dynamic and changing nature of the projects for MLESP. Through a meticulous process of DSR, ME, and PA, we formulate a hybrid PM method, method base and an ME framework. This approach embraces different knowledge domains as an opportunity for innovation in PM for ML. We believe that this study represents a promising and initial effort to integrate and enhance the SE and PM processes for ML applications within various domains.

It is also important to note some research directions for future studies as follows:

- Customization for diverse domains: ML projects span a wide spectrum of domains, from healthcare to finance. Future research should focus on tailoring the PM framework to suit specific domain requirements and challenges, ensuring its applicability across various sectors.
- Comparative studies: Comparative studies that evaluate the hybrid PM method and the ME framework against other tailoring methods and approaches can provide valuable insights into its strengths and weaknesses, and they can inform further refinements.
- Continuous improvement: While comprising the principles of agile methods, the hybrid PM method should continuously evolve and adapt. Regular updates and improvements should be based on feedback from practitioners and insights from ongoing research.
- Tooling and automation: The development of supporting tools and platforms that automate various aspects of the ME and PM framework, and the hybrid PM method can enhance its usability. Investigating the feasibility of such tools and their integration into the ML process can be an important future direction.
- Education and training: Preparing project managers and teams for the challenges of complex projects is essential. Future research can explore the development of training programs and educational materials that align with the principles of the proposed ME framework and the hybrid PM method.

In conclusion, ML, PM, ME, and PA knowledge domains are integrated to tackle the evolving challenges of complex and data-driven ML projects. Our intention to engineer a robust and adaptable PM method for ML led us to develop an ME framework for PM rooted in ME principles and environments. This framework integrates the complexities of ML with well-known PM methods and SE practices. It also provides a structured yet flexible approach to address the complexities of data-driven projects. Therefore, this study has the potential to contribute not only to ML PM and BUHA but also to advancing process management within mission and safety-critical domains like healthcare. By addressing current limitations and accommodating the needs of the proposed solution, we intend to inspire researchers and practitioners with an effective toolset for a successful PM. The future holds research prospects for the application of this approach to ML PM in diverse contexts.

Acknowledgement

This study is an extension of a research project supported by Baskent University (Project ID: 2021-11-0032), titled "An enterprise architecture and artificial intelligence implementation roadmap for Baskent University Hospital Ankara". The author confirms that he has no competing financial and other types of interests, or personal and organizational relationships that could have appeared to influence the study reported in this paper.

Appendices

Table A.1. ME elements of ML

#	List of Tasks	List of Roles	List of Work Products
1	Problem definition	ML project manager	ML training data
2	Data acquisition	ML engineer	ML test data
3	Data processing	Data engineer	ML training model
4	Feature engineering	Data scientist	ML pipeline
5	Exploratory data analysis		ML deployed model
6	Model training and testing		
7	Model evaluation and validation		
8	Model deployment		
9	Performance monitoring		

Table A.2. ME elements of CRISP-DM

#	List of Tasks	List of Roles	List of Work Products
1	Business understanding	Not defined	Project plan
2	Data understanding	-	Data description report, data exploration report
3	Data preparation	-	Data quality report, data cleaning report
4	Modeling	-	Learning model, test plan
5	Evaluation	-	Deployment plan
6	Deployment	-	Monitoring and maintenance plan
7	-	-	Final report

Table A.3. ME elements of TDSP

#	List of Tasks	List of Roles	List of Work Products
1	Business understanding	Project manager	Charter document
2	Data acquisition and understanding	Project lead	Data dictionary
3	ML modeling		
3.1	Feature engineering	Solution architect	Data quality report
3.2	Model training	Data scientist	ML Pipeline
3.3	Model evaluation	Data engineer	Solution architecture
4	ML deployment	Application developer	Exit report

Table A.4. ME elements of Scrum

#	List of Tasks	List of Roles	List of Work Products
1	Product backlog and prioritizing	Product owner	Product backlog
2	Release planning	Scrum master	Sprint backlog
3	Sprint planning	Scrum team	Task board
4	Sprint execution		Sprint burndown chart
5	Daily scrum		Potentially shippable product increment

6	Sprint review		Release burndown chart
7	Sprint retrospective		

Table A.5. ME elements of Kanban

#	**List of Tasks**	**List of Roles**	**List of Work Products**
1	Standard software development processes	Defined if required	Kanban task board
#	**Kanban Principles**		
1	Visualize		
2	Limit work in process		
3	Manage flow		
4	Make process policies explicit		
5	Implement feedback loops		
6	Improve collaboratively and evolve experimentally		

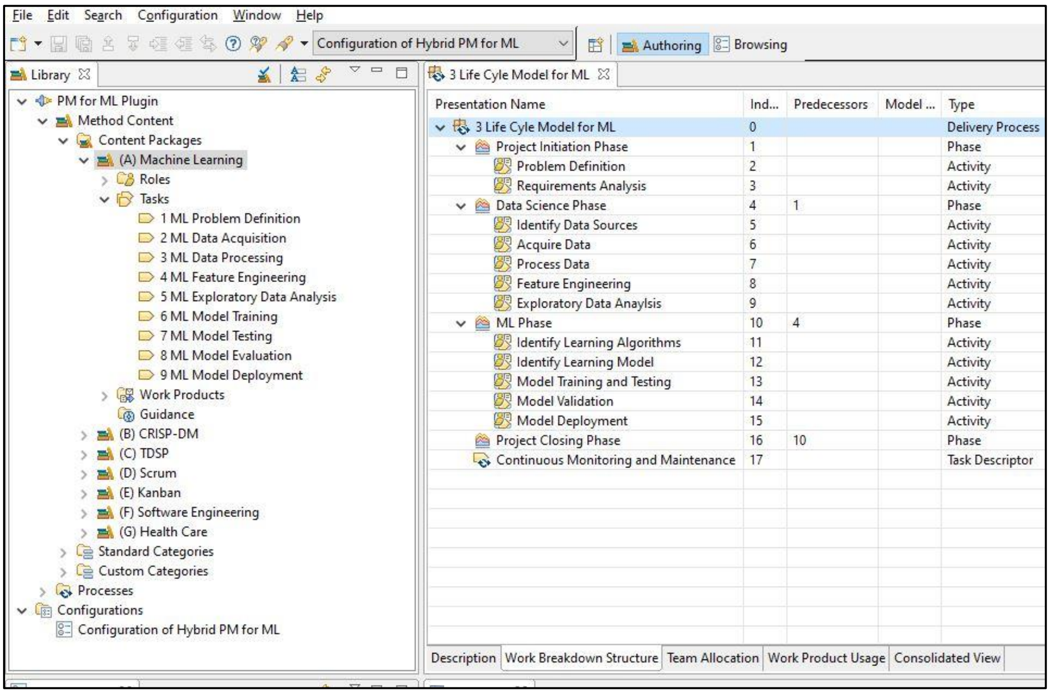

Figure A.1. ME elements of ML in method base.

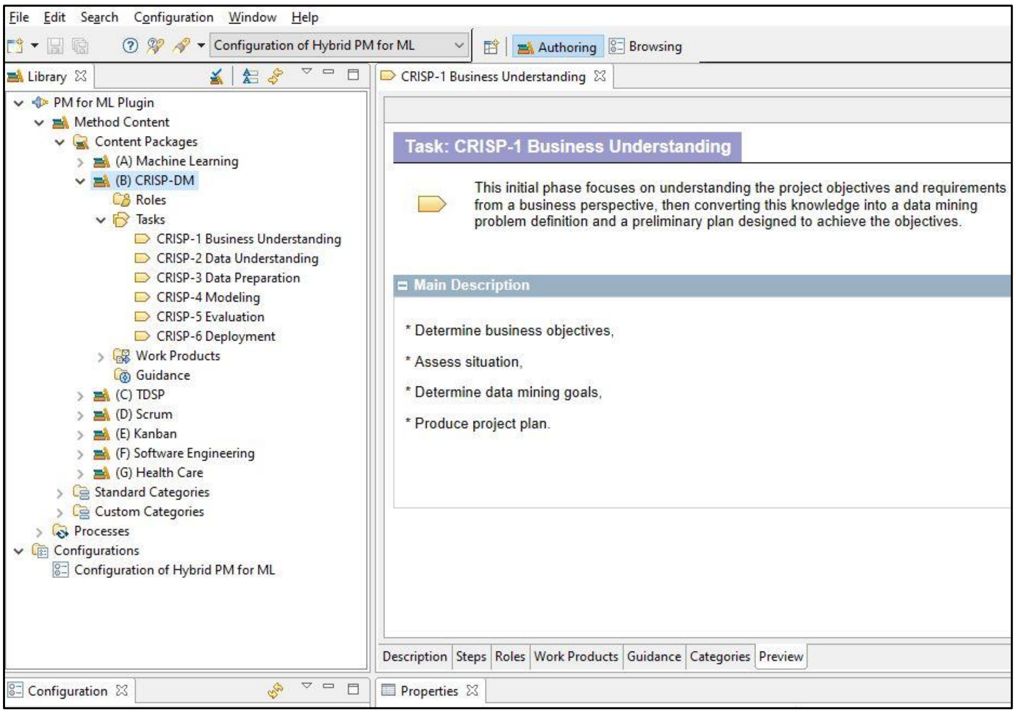

Figure A.2. ME elements of CRISP-DM in method base.

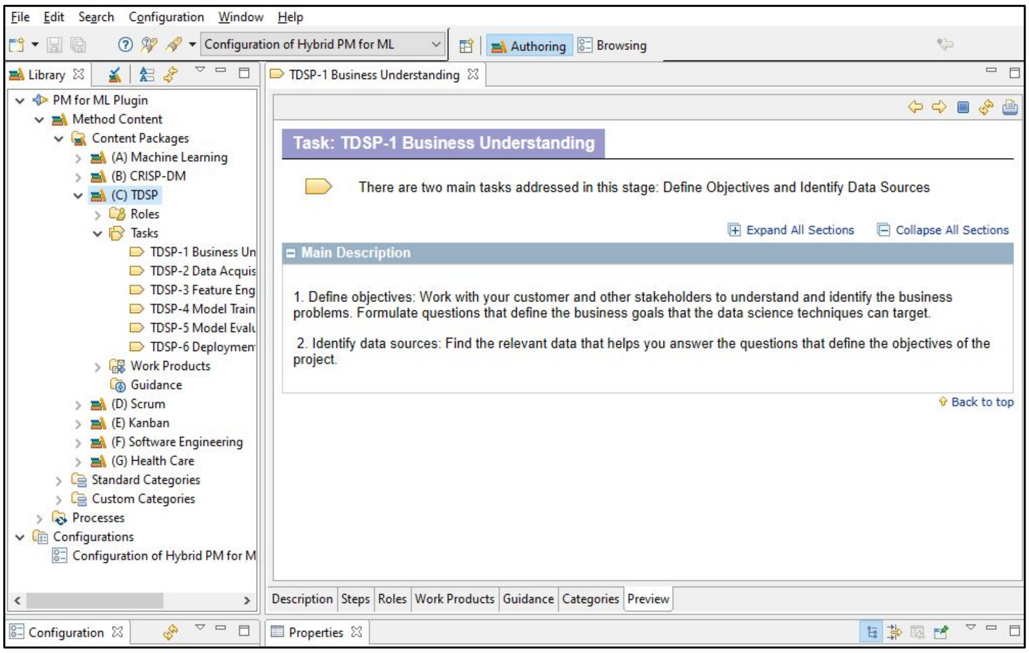

Figure A.3. ME elements of TDSP in method base.

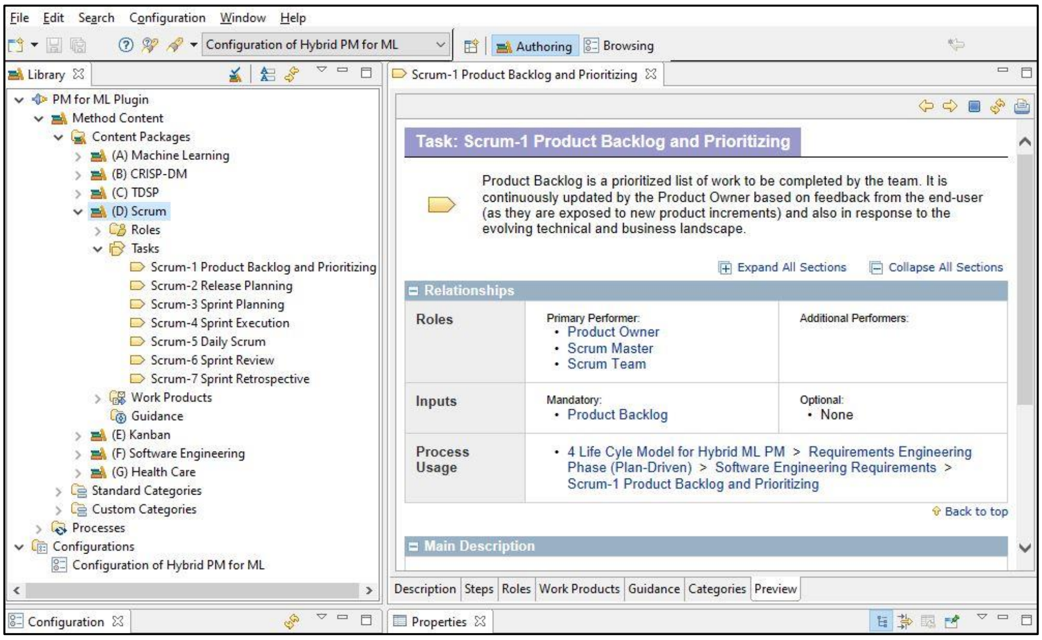

Figure A.4. ME elements of Scrum in method base.

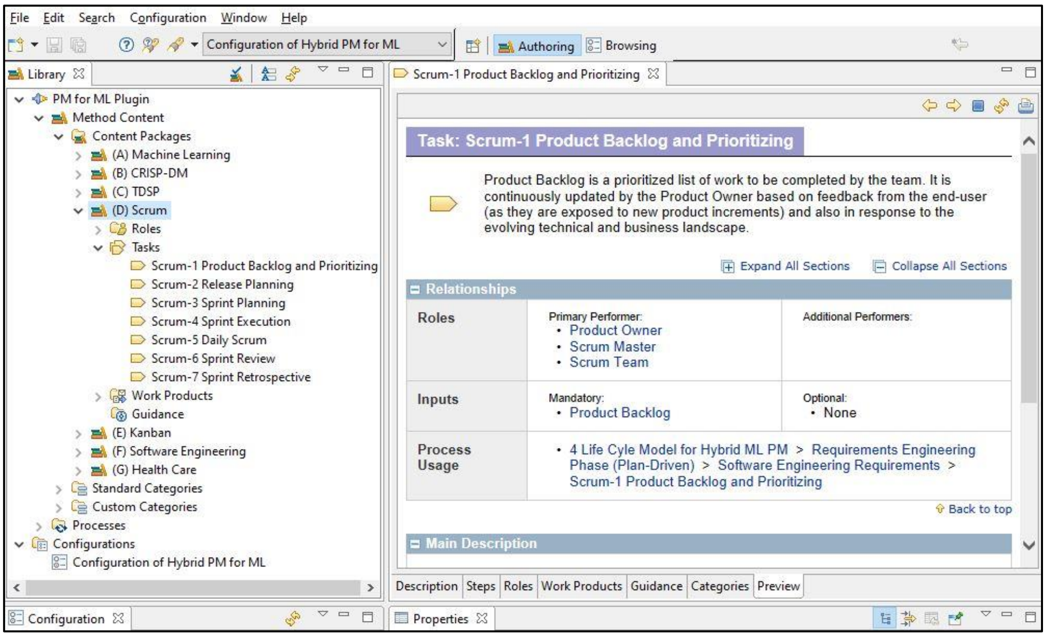

Figure A.5 ME elements of Kanban in method base.

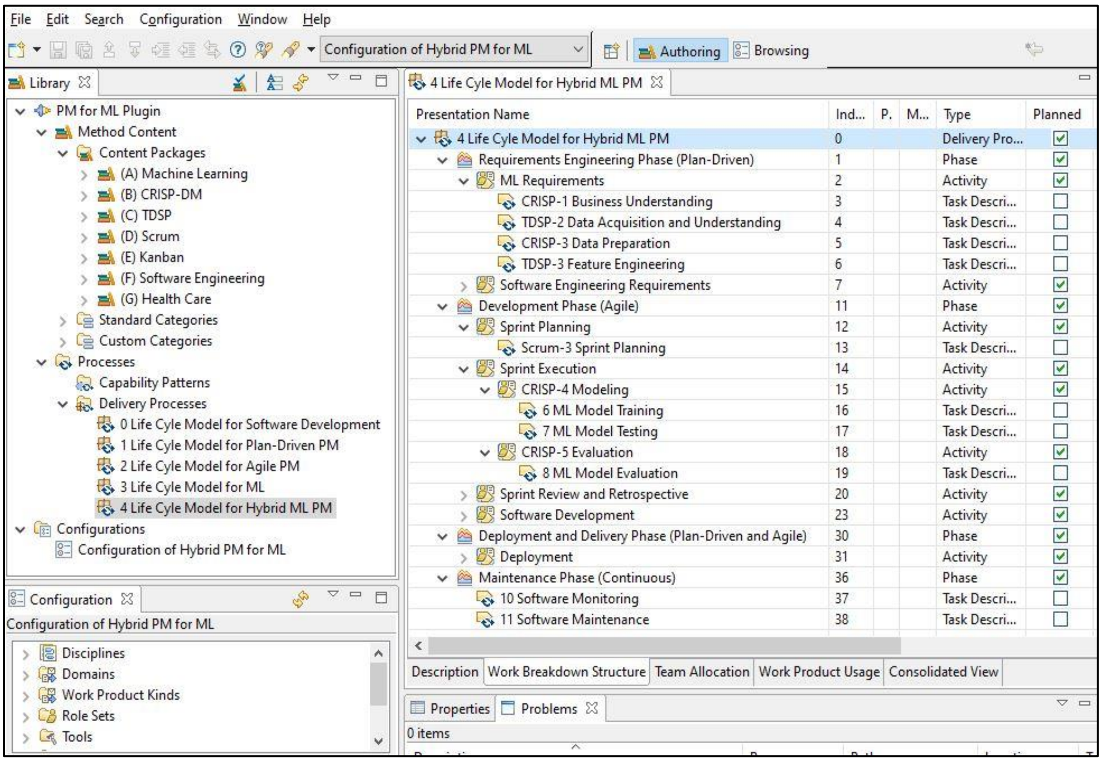

Figure A.6 The method base and the new hybrid PM method for MLESP.

References

1. Bughin J. & Hazan E. (2017). Five management strategies for getting the most from AI. *MIT Sloan Management Review*. "https://sloanreview.mit.edu/ article/five-management-strategies-for-getting-the-most-from-ai".
2. Khomh, F., Adams, B., Cheng, J., Fokaefs, M. & Antoniol G. (2018). Software engineering for machine-learning applications: The road ahead. *IEEE Software*, *35*(5), 81-84.
3. Giray, G. (2021). A software engineering perspective on engineering machine learning systems: State of the art and challenges. *The Journal of Systems & Software*, *180*, 1-35.
4. Oun T. A., Blackburn T. D., Olson B. A. & Blessner P., (2016). An enterprise-wide knowledge management approach to project management. *Engineering Management Journal*, *28*(3), 179-192, DOI: 10.1080/10429247.2016.1203715.
5. Haumer, P. (2007). Eclipse process framework composer part 1: Key concepts & Part 2: Authoring method content and processes, Eclipse Foundation, "http://www.eclipse.org/epf/ general/ EPFComposerOverviewPart1.pdf".
6. Uysal M.P. (2023). Toward a method engineering framework for project management and machine learning. In *IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*, Torino, *Italy*.
7. Saltz, J., Shamshurin, I. & Crowston K (2017), Comparing data science project management methodologies via a controlled experiment. In *Proceedings of the 50th Hawaii International Conference on System Sciences*, 1013-1022.
8. Uysal M.P. (2022). An integrated and multi-perspective approach to the requirements of machine learning. *In Proceedings of IFIP International Conference on Industrial Information Integration (ICIIIE 2022), Bangkok, Thailand*.
9. Saltz, J. (2015). The need for new processes, methodologies and tools to support big data teams and improve big data project effectiveness. In *Proceedings of the IEEE International Conference on Big Data*, 2066-2071.
10. Uysal M.P. (2021). *Machine learning and data science project management from an agile perspective.* In V. Naidoo and R. Verma (Eds), Methods and challenges in contemporary challenges for agile project management, IGI Global, NY, USA, 73-88.
11. Sellers, B.H. Ralyté, J., Agerfalk, P.J. & Rossi,M. (2014). *Situational method engineering. Springer*, NY, USA.
12. Amershi, S., Begel, A. C., Bird, R. DeLine, H. Gall, Kamar, E., Nagappan, N., Nushi B. & Zimmermann, T. (2019). *Software engineering*

for machine learning: A case study, Microsoft Research, https://www.microsoft.com.
13. Campanelli, A.S., Parreiras & F.S., (2015). Agile methods tailoring-A systematic literature review, *The Journal of Systems and Software*, 110, 85-100.
14. Kumeno, F. (2019). Software engineering challenges for machine learning applications: A literature review. *Intell. Decis. Technol.*, *13* (4), 463-476.
15. Bourque, P. & Richard, E. (2014). SWEBOK Version 3.0, IEEE, ISBN-10: 0-7695-5166-1.
16. Lwakatare, L.E., Raj, A., Crnkovic, I., Bosch J. & Olsson H.H. (2020). Large-scale machine learning systems in real-world industrial settings: A review of challenges and solutions. *Information and Software Technology*, *127*, 106368.
17. Nascimento, E., Nguyen-Duc, A., Sundbø I. & Conte T. (2020). Software engineering for artificial intelligence and machine learning software: A systematic literature review. *ArXiv*, arXiv:2011.03751.
18. Fernández, S.M., Bogner, J. Franch, X., Oriol, M., Siebert J. A. Trendowicz, Vollmer, A.M. & Wagner, S. (2022). Software engineering for AI-Based systems: A survey. *ACM Transactions on Software Engineering and Methodology*, *31* (2), 1-59.
19. Sithambaram, E. (2018). *Identifying pitfalls in machine learning implementation projects: A case study of four technology-intensive organizations*. Unpublished MS thesis, Royal Institute of Technology, Sweden.
20. Rahman S., Rivera E., Khomh F., Guéhéneuc Y.G. & Lehnert B. (2019). Machine learning software engineering in practice: An industrial case study. *ArXiv*, https://arxiv.org/abs/1906.07154.
21. Ishikawa, F. & Yoshioka, N. (2019). How do engineers perceive difficulties in engineering of machine-learning systems? In *Proceedings of IEEE/ACM 6th International Workshop on Software Engineering Research and Industrial Practice*.
22. Gupte, A. (2018). *Determining critical success factors for big data projects*. Unpublished Doctoral Dissertation, Purdue University, USA.
23. Tsoy, M. & Staples, D.S. (2020). Exploring critical success factors in agile analytics projects. In *Proceedings of 53rd Hawaii International Conference on System Sciences*, 984-993.
24. Kuhrmann, M., Diebold, P., Münch, J., Tell, P. Garousi, V., Felderer, M. Trektere, K., McCaffery, F., Linssen, O., Hanser, E. & Prause, C.R. (2017). Hybrid software and system development in practice: Waterfall, Scrum, and beyond. In *Proceedings of International Conference on Software System Process*.
25. Gill, A.Q. Sellers, B.H. & Niazi, M. (2018). Scaling for agility: A reference model for hybrid traditional-agile software development methodologies. *Inf. Syst. Front*, *20*, 315-341.
26. Papadakis, E. & Tsironis, L. (2020). Towards a hybrid project management framework: A systematic literature review on traditional, agile and hybrid techniques. *The Journal of Modern Project Management*, *8* (2) 124-139.
27. Conforto, E.C. & Amaral, D.C. (2016). Agile project management and stage-gate model. A hybrid framework for technology-based companies. *Journal of Engineering and Technology Management*, *40*, 1-24.
28. Zasa, F.P. Patrucco, A. & Pellizzoni, E. (2021). Managing the hybrid organization: how can agile and traditional project management coexist? Research-Technology Management, *64* (1) 54-63.
29. Sithambaram, J., Nasir, M.H.N.B.M. & Ahmad R., (2021). Issues and challenges impacting the successful management of agile-hybrid projects: A grounded theory approach. *International Journal of Project Management*, "https://doi.org/ 10.1016/j.ijproman. 2021.03.002".
30. Azenha, C.F., Reis, D.A. & Fleury A.L. (2021). The role and characteristics of hybrid approaches to project management in the development of technology-based products and services. *Project Management Journal, 52* (1), 90-110.
31. Uysal M.P., (2022). Machine learning-enabled healthcare information systems in view of industrial information integration engineering. *Journal of Industrial Information Integration*, *30*, (1), 100382.
32. Sellers, B. H., France R., Georg, G. & Reddy R. (2007). A method engineering approach to developing aspect-oriented modelling processes based on the OPEN process framework. *Information and Software Technology, 49*, 761-773.
33. SPEM (2008). Software & Systems Process Engineering Meta-Model Specification v 2.0. *OMG Technical Document*, USA.
34. EPF (2023). Eclipse process framework project, Eclipse Foundation, "https://projects.eclipse. org/projects/ technology.epf".
35. Haumer P. (2007). Eclipse process framework composer Part 1: Key concepts & Part 2: Authoring method content and processes, *Eclipse Foundation*, "http://www.eclipse.org/epf/ general/ EPFComposerOverviewPart1.pdf".
36. Bergstra, J.A., Ponse, A. & Smolka, S.A. (2001). *Handbook of process algebra*. Elsevier Science, Netherlands.
37. Baeten, J.C.M (2004). A brief history of process algebra. *Theoretical Computer Science, 335*, 131-146.
38. Bergstra, J.A. & Klop, J.W. (1985). Algebra of communicating processes with abstraction. *Theoretical Computer Science, 37*, 77-121.
39. Luttik, B. (2006). What is algebraic in process theory? *Electronic Notes in Theoretical Computer Science*, *162*, 227-231.

40. Fokkink, W. (2007). *Introduction to Process Algebra*, Springer, USA.
41. CRISP-DM (2000). *CRISP-DM 1.0: Step-by-step data mining guide*. The CRISP-DM consortium, SPSS Publication, USA.
42. V. Plotnikova, M. Dumas & F.P. Milani (2022) Applying the CRISP-DM data mining process in the financial services industry: Elicitation of adaptation requirements. *Data & Knowledge Engineering*, *139*, 102013.
43. TDSP, (2024). What is the team data science process? "https://docs.mcrosoft.com".
44. Sutherland, J. & Schwaber K., (2017). The Scrum Guide, "http://www.scrumguides.org".
45. Hammarberg, M. & Sunden, J. (2014). *Kanban in action.* Manning Publications, USA.
46. DDS, (2023). Data driven scrum framework, "https://www.datascience-pm.com/data-driven-scrum".
47. Gemino, A., Reich, B.H. & Serrador, P.M (2021). Agile, traditional, and hybrid approaches to project success: Is hybrid a poor second choice? *Project Management Journal, 52* (2), 161-175.
48. Hevner, A. March, S., Park, J. & Ram S. (2004). Design science in information systems research. *MIS Quarterly, 28* (1), 75-105.
49. S. Shafiee, Y. Wautelet, S. Poelmans & S. Heng (2023). An empirical evaluation of scrum training's suitability for the model-driven development of knowledge-intensive software systems. *Data & Knowledge Engineering, 146*, 102195.
50. Conboy, K. & Fitsgerald, B (2010). Method and developer characteristics for effective agile method tailoring: A study of XP expert opinion. *ACM Transactions on Software Engineering and Methodology, 20* (2), 1-30.
51. Haakman, M., Cruz, L., Huijgens, H. & Deursen A. (2021). AI lifecycle models need to be revised: An exploratory study in Fintech. *Empirical Software Engineering, 26* (95), 1-29.
52. Ramasamy, D., Sarasua, C., Bacchelli, A. & Bernstein, A. (2023). Workflow analysis of data science code in public GitHub repositories. *Empirical Software Engineering, 28* (7), 1-47.
53. Gill, A.Q., Sellers, B.H. & Niazi, M. (2018). Scaling for agility: A reference model for hybrid traditional-agile software development methodologies. *Inf. Syst. Front., 20*, 315-341.
54. Abro, A.A., Siddique, W.A., Talpur, M.S.H., Jumani, A.K. & Yaşar E. (2023). A combined approach of base and meta learners for hybrid system. *Turkish Journal of Engineering, 7*(1), 25-32.
55. Maza, D., Ojo, J. O., & Akinlade, G. O. (2024). A predictive machine learning framework for diabetes. *Turkish Journal of Engineering*, *8*(3), 583-592, "https://doi.org/10.31127/tuje.1434305".
56. Sinap, V. (2024). Comparative analysis of machine learning techniques for credit card fraud detection: Dealing with imbalanced datasets. *Turkish Journal of Engineering, 8*(2), 196-208, "https://doi.org/ 10.31127/tuje.1386127".
57. Leka, B., & Hoxha, K. (2024). Software engineering methodologies in programming companies in Albania. *Engineering Applications, 3*(1), 85–91, "https://publish.mersin.edu.tr/index.php/enap /article/view/1506".
58. Kayıran, H. F. (2022). The function of artificial intelligence and its sub-branches in the field of health. *Engineering Applications, 1*(2), 99–107, "https://publish.mersin.edu.tr/index.php/enap /article/view/328".
59. Dirik, M. (2023). Machine learning-based lung cancer diagnosis. *Turkish Journal of Engineering, 7*(4), 322-330, "https://doi.org/10.31127/tuje.1180931".
60. Juraev, D. A., & Bozorov, M. N. (2024). The role of algebra and its application in modern sciences. *Engineering Applications, 3*(1), 59–67, "https://publish.mersin.edu.tr/index.php/enap /article/view/1499".
61. İncekara, Ç. Ö. (2023). Industrial internet of things (IIoT) in energy sector. *Advanced Engineering Science*, 3, 21–30. "https://publish.mersin.edu.tr/ index.php/ades/article/view/839".
62. Mema, B. & Basholli , F. (2023). Internet of Things in the development of future businesses in Albania. *Advanced Engineering Science*, 3, 196–205. "https://publish.mersin.edu.tr/index.php/ades /article/view/1325".
63. Grant, D., & Mergen, A. E. (2013). Using SPC in conjunction with APC. *Quality Engineering*, *23* (4),360-364.
64. Grant, D., & Mergen, A. E. (2009). Towards the use of Six Sigma in software development. *Total Quality Management & Business Excellence*, *20*(7), 705–712.