Formalizing PI Planning in Scaled Agile with Model-Driven Engineering: A Metamodel and Framework for Strategic Alignment

Flavien Hervé SOMDA, Désiré GUEL, Kisito Kiswendsida KABORE

Université Joseph KI-ZERBO (U-JKZ), Burkina Faso

*E-mail : flavien.somda@ujkz.bf

Abstract

Program Increment (PI) Planning is a critical synchronization event in scaled Agile methodologies, yet remains largely informal, with limited analytical rigor or strategic alignment. This paper introduces a formal metamodel and a model-driven framework to support structured PI Planning in large-scale software development. Our metamodel captures essential planning artifacts, including features, teams, iterations, risks, and dependencies, and is enriched with OCL constraints for validation and traceability. We extend the model to align planning with business value streams and introduce stakeholder-specific viewpoints for tailored analysis. Compared to tools like Jira or Mural, our approach provides formal semantics, automated reasoning, and extensibility. This work lays the foundation for a Domain-Specific Language (DSL) for PI Planning, bridging the gap between Agile practice and formal modeling.

Keywords

Program Increment Planning, Agile Methodologies, Metamodel, Model-Driven Engineering, Impact Analysis, Modeling Framework

I INTRODUCTION

Program Increment (PI) Planning is a cornerstone of scaled Agile methodologies such as the Scaled Agile Framework (SAFe), where multiple Agile teams must align on shared goals, synchronize work, and deliver value within fixed timeboxes. While PI Planning is designed to foster collaboration and visibility across teams, it remains a challenging process to execute effectively, particularly in large-scale software development environments. Among the main obstacles are the complexity of managing inter-team dependencies, capacity constraints, risk factors, and the alignment of development efforts with evolving business priorities.

While widely adopted tools support collaboration and visualization, they lack semantic rigor and fail to encode structural constraints or traceability paths. Consequently, they fall short in enabling automated reasoning, validation, or cross-cutting analysis—capabilities increasingly expected in critical planning processes at scale.

In this context, Model-Driven Engineering (MDE) offers promising opportunities to formalize and improve the PI Planning process. By using models as central artifacts, MDE can provide high-level abstractions, structural consistency, and automation features that enhance planning, analysis, and decision-making. However, to the best of our knowledge, no dedicated metamodel has yet been proposed for capturing the specific semantics and structure of PI Planning events.

This paper proposes a formal metamodel and a conceptual framework for modeling and analyzing PI Planning. The contribution is positioned as a foundational step toward the development of a Domain-Specific Language (DSL) tailored to this critical Agile process. Our metamodel captures the structural components and constraints involved in PI Planning and introduces an extension for linking planning artifacts to business value streams. The associated framework enables multiple stakeholder viewpoints and supports automated impact analysis.

By grounding PI Planning in formal modeling through MDE, this research contributes to reconciling the current disconnect between industry practice and the analytical rigor demanded by software engineering research. We hypothesize that such formalization enhances planning reliability, supports automated analysis, and improves strategic alignment—a claim to be tested in future empirical work.

The main contributions of this paper are:

- A formal metamodel capturing the core and strategic elements of PI Planning.
- An extensible viewpoint-driven framework for stakeholder-specific analysis and visualization.
- A comparative evaluation with existing tools, highlighting the added value in traceability, validation, and strategic alignment.
- A roadmap toward a dedicated DSL for PI Planning, integrating MDE-based automation.

The remainder of this paper is structured as follows. Section II reviews the relevant literature on Agile methodologies, Program Increment (PI) Planning, and Model-Driven Engineering (MDE). Section III provides an overview of current PI Planning practices based on empirical observations. Section IV defines the research problem, objectives, and methodological approach. Section V presents the core metamodel for PI Planning and its extension for business alignment. Section VI introduces the conceptual MDE-powered framework and demonstrates its analytical capabilities. Section VII discusses the theoretical and practical implications, and Section VIII concludes the paper with a summary of contributions and future research directions.

II RELATED WORK

In this section, we provide a comprehensive review of the existing literature pertaining to Agile methodologies, Program Increment (PI) Planning, and modeling techniques. By examining prior research and publications, we establish the context and foundation upon which our metamodel and framework are built. The literature review is organized into key thematic areas:

2.1 Agile Methods and Scaled Frameworks

The utilization of agile principles for software development in various industries has been on the rise ever since the Agile Manifesto was published in 2001 [1, 2]. Even though cultural and organizational factors affect the adoption of agile approaches, their adoption continues to grow, as agility remains a crucial competence for navigating change [3, 4]. Agile Methods, characterized by their iterative and customer-centric processes [5], have been used to build High-Performance Teams [6], i-e, teams that emerge when individuals within the group depend on one another, align their actions with a shared vision, foster open communication to facilitate the growth of their endeavors, cultivate trust, and embrace collective leadership, thus fostering innovation through the unique contributions of each team member. As an effect, agile methods have contributed to lower software development failures and improve quality [7] [8]. The involvement of Customer decreases the chance of software rejection in the last phases [9].

2.2 Program Increment (PI) Planning: Gaps and Challenges

Program Increment (PI) Planning is a key event in the Scaled Agile Framework (SAFe). It aims to synchronize multiple Agile teams around shared objectives. Although widely adopted in large-scale organizations, PI Planning remains a highly manual and coordination-intensive activity. It often relies on virtual whiteboards (e.g., Mural, Klaxoon) or backlog management tools (e.g., Jira, Azure Boards) continuing to exacerbate the gap that persists between agile practice and research [2].

In fact, while practical, the tools offer limited support for formal modeling. They do not capture the semantics of planning artifacts such as teams, features, dependencies, or risks in a structured and analyzable way. For instance, dependencies are often visualized using post-it arrows without semantic validation, and overcommitments must be manually calculated.

To the best of our knowledge, no existing work proposes a dedicated metamodel that formalizes the structure and constraints of PI Planning in a way that enables automated analysis, multi-view modeling, and strategic alignment assessment. This research addresses that gap by introducing an ecore-based metamodel enriched with OCL constraints, extended to include value stream elements, and coupled with a viewpoint-driven framework tailored to the roles and perspectives of key stakeholders.

2.3 Model-Driven Engineering in Agile Contexts

Model-Driven Engineering (MDE) is an established approach in software engineering that uses models as central artifacts for system specification, validation, simulation, code generation, testing and maintenance, among various other activities [10][11]. Key benefits of MDE include abstraction, consistency enforcement, reusability, automation, traceability, visualization, and domain-specific modeling. Prior studies demonstrate how metamodeling can support high-level representations of complex processes, and helps in system development [12]. These principles are particularly relevant to Agile at scale, where coordination, visibility, and adaptability are essential. The array of advantages offered by Model-Driven Engineering encompasses the following:

- **Abstraction and Simplification:** Metamodeling allows the creation of high-level, abstract representations of complex systems and processes [13]. This abstraction simplifies the modeling process and makes it easier to understand and communicate about complex systems.
- **Consistency and Reusability:** Metamodels enforce consistency by defining a set of rules and constraints for models created based on them. This ensures that models conform to established standards and best practices [14]. Additionally, metamodels promote reusability, as they can serve as templates for creating similar models in different contexts.
- **Automation:** MDE enables automation through code generation [13] [15]. By defining models and their relationships in a metamodel, you can automatically generate code, documentation, or other artifacts. This reduces the likelihood of errors and speeds up development.
- **Traceability:** MDE enables traceability [16] by establishing relationships between different elements in models. This traceability helps in understanding the impact of changes and in managing complex systems with multiple dependencies.
- **Visualization and Documentation:** Metamodels often come with visualization tools that help developers and stakeholders better understand and visualize complex systems. Ad-

ditionally, the structured nature of metamodels supports automatic documentation generation [15].

- **Domain-Specific Modeling:** Metamodeling allows the creation of domain-specific modeling languages [13] tailored to organization's needs. This enables developers to work at a higher level of abstraction, focusing on domain-specific concepts rather than low-level technical details.

To our knowledge, no existing work integrates formal metamodeling, OCL-based validation, and stakeholder-specific views to support end-to-end PI Planning analysis and alignment. This gap is precisely what our work addresses.

III CURRENT PRACTICES IN PI PLANNING EVENT

While the specific sequence of activities within a PI Planning Event may naturally differ from one organization to another, informal discussions and participation in a few Program Increment Planning events have revealed that certain key activities are consistently present in most PI Planning events:

1. **Business Context and Overview:**
   - Leadership provides a high-level overview of the business context, market dynamics, and strategic themes.
   - The organization's vision, mission, and goals are discussed to ensure alignment.
2. **Pre-Planning:**
   - Teams and individuals have already conducted pre-planning activities to review their backlogs and prepare for the PI Planning event.
3. **PI Objectives and Team Breakouts:**
   - PI Objectives are defined, outlining the specific outcomes and goals for the upcoming Program Increment.
   - Teams break out into separate sessions, often organized by Agile Release Trains (ARTs), to create their plans.
   - Teams collaborate to select and commit to the features, stories, and work items they will complete during the PI.
4. **Dependencies and Risks:**
   - Teams identify and document dependencies between their work items and other teams' work. A visual example of this phase is demonstrated in the Fig 1 figure, which highlights the interdependencies among various team activities.
   - Risks and impediments are raised and discussed, with action plans to mitigate them.
5. **Consolidation:**
   - Teams finalize their PI plans, ensuring that all dependencies are documented and understood.
   - Adjustments are made as needed to accommodate any changes that arose during the event.
6. **Confidence Vote:**
   - After the breakouts and discussions, teams often conduct a confidence vote to assess their collective confidence in their PI plan. This is a way to gauge the team's alignment and understanding of the plan.
7. **Closing and Commitment:**
   - The PI Planning event concludes with a commitment to the plan. Teams express their commitment to achieving the PI objectives.
   - Key takeaways and action items are summarized, and the event is officially closed.

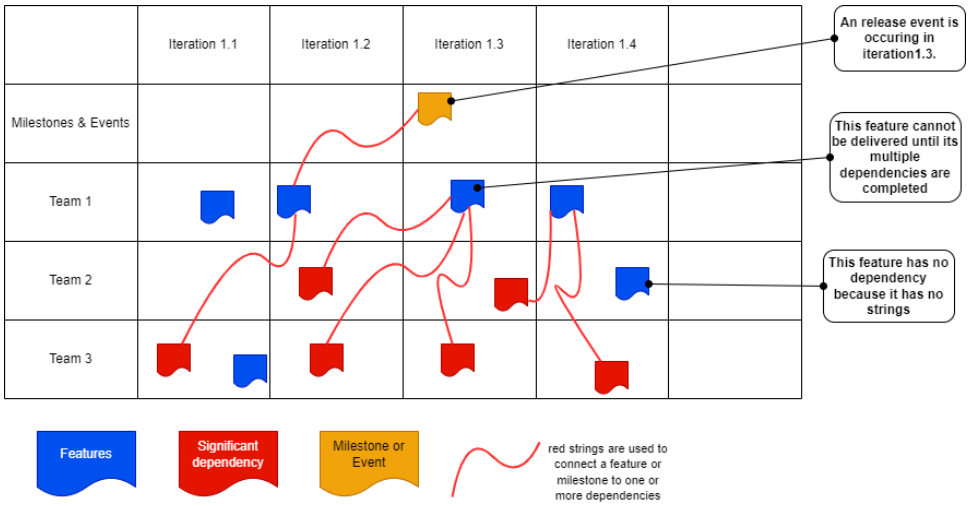

Figure 1: Sample artefact illustrating dependencies in PI Planning

IV PROBLEM STATEMENT AND RESEARCH OBJECTIVES

4.1 Limitations of Current PI Planning Tools

PI Planning has evolved with the introduction of digital collaboration platforms. However, these tools often lack structured guidance and formal representations. Challenges include:

- Lack of traceability and standardization in dependency management;
- Inability to validate team capacity constraints or strategic alignment automatically;
- Manual tracking of risks and changes;
- Limited integration between PI artifacts and organizational goals.

4.2 Research Questions

To guide this study and structure our contribution, we define the following research questions:

Research Questions

1. **RQ1**: How can Program Increment (PI) Planning be formally represented to improve structural consistency, traceability, and strategic alignment in Agile-at-scale contexts?
2. **RQ2**: What metamodel elements and formal constraints are essential to accurately capture the semantics and rules of PI Planning?
3. **RQ3**: How can a model-driven framework support stakeholder-specific views and facilitate automated validation and analysis during the planning process?
4. **RQ4**: To what extent does the proposed approach offer advantages over existing tools in terms of analytical capabilities and organizational alignment?

4.3 Mapping of Research Questions to Paper Sections

The following table summarizes where each research question is primarily addressed in the paper.

Table 1: Mapping of Research Questions to Sections

Research Question	Relevant Section(s)
RQ1	Section V – Formalization of PI Planning
RQ2	Section V.2 – Formal Metamodel for PI Planning
RQ3	Section VI – Model-Driven Framework for Stakeholder-Centric PI Planning
RQ4	Section VII – Discussion and Comparison with Existing Approaches

4.4 Methodology

4.4.1 Research Approach

Our approach is based on the design of a formal metamodel using Ecore and OCL. We used the Eclipse Modeling Framework (EMF) and Sirius to define structural concepts, constraints, and graphical representations aligned with Agile PI Planning events. The modeling was driven by observed practices, challenges, and needs in large-scale Agile environments.

4.4.2 Data Sources and Observations

The design of our metamodel was informed by informal field observations and active participation in four PI Planning events within a large banking organization. These events involved four to five Agile teams each, providing insights across multiple teams and coordination patterns. These observations offered valuable insight into recurrent patterns, roles, and coordination challenges during PI Planning events. While the current work remains conceptual, it reflects realistic scenarios encountered in practical settings.

4.5 Research Contribution

This paper proposes a metamodel and framework grounded in MDE to address the above limitations. Our contribution includes:

- A formal core metamodel for PI Planning;
- An extension for value stream alignment;
- Viewpoint-based framework enabling traceability and impact analysis;
- Foundations for a future Domain-Specific Language (DSL) for PI Planning.

V FORMALIZATION OF PI PLANNING

5.1 Motivations for a Formal Model-Based Approach to PI Planning

PI Planning, a cornerstone practice in Agile at scale frameworks such as the Scaled Agile Framework (SAFe) [17], has seen a significant shift in recent years with the adoption of digital collaborative tools like Mural or klaxoon. While these tools offer flexibility and virtual collaboration, they also present unique challenges when compared to traditional, formalized planning methods. One of the primary challenges lies in the absence of a formal model or framework to guide the planning process. Here are some key difficulties that organizations may encounter:

- **Lack of Structured Guidance:** Traditional PI Planning often relies on structured agendas and predefined templates to ensure consistency and alignment. Collaborative tools like Mural, while versatile, may lack this level of structured guidance. As a result, teams may struggle to maintain a consistent and standardized planning process.
- **Dependency Management:** PI Planning involves identifying and managing dependencies between teams and work items. Without a formal model, dependency tracking can become more complex and error-prone in collaborative tools. Teams may find it challenging to visualize and address intricate interdependencies effectively.
- **Visibility and Traceability:** In a formal PI Planning model, there is often built-in visibility into the progress of work items and their alignment with strategic objectives. Collaborative tools might not provide the same level of traceability, making it harder to assess whether the PI plan is on track to meet its objectives.
- **Alignment Challenges:** Achieving alignment across teams and ARTs (Agile Release Trains) is a core goal of PI Planning. In the absence of a formal model, it can be more challenging to ensure that all teams are aligned with the same priorities and objectives, potentially leading to misalignment.
- **Documentation and Reporting:** Formal models often support automatic documentation and reporting, which can be lacking in collaborative tools. Generating accurate and up-to-date documentation may require manual effort.
- **Integration with Other Tools:** Collaborative tools need to integrate seamlessly with other software used in the organization's Agile process. Without a formal model to guide integration, ensuring data consistency and accuracy can be challenging

Our objective in furnishing a metamodel for PI Planning is to establish a foundational framework upon which it becomes feasible to construct more organized and cohesive layers, facilitating the planning methodology and housing the data it generates. As an illustration, the metamodel could facilitate the creation of perspectives and representations, such as tables and diagrams, which can be employed to model planning results and automatically identify discrepancies between Agile teams' capacities and work items allocated to them during a specific iteration.

5.2 Formal Metamodel for PI Planning

This section presents the core metamodel designed to formally represent the key structural elements of a Program Increment (PI) Planning event. The metamodel is defined using UML class diagrams and incorporates Object Constraint Language (OCL) rules to enforce consistency between elements.

To illustrate the motivation behind our model, consider the following scenario:

A PI Planning event involves three Agile teams (Team A, B, and C). Each team commits to delivering a subset of features, broken down into stories distributed across iterations. Some stories have dependencies on other teams' work, and business stakeholders need to ensure that these features align with the company's quarterly strategic objectives. Furthermore, each team has limited capacity and must avoid overcommitment. Risks need to be tracked and visualized across iterations.

Our metamodel aims to capture the structural components necessary to represent such a scenario, including:

- **Teams**, **Features**, and **Stories**, linked via assignment and decomposition;
- **Iterations** and **TeamCapacity** constraints;

- **Dependencies** between stories (internal and external);
- **Risks** associated with planning elements;
- Validation constraints such as "a team's total story points per iteration must not exceed capacity."

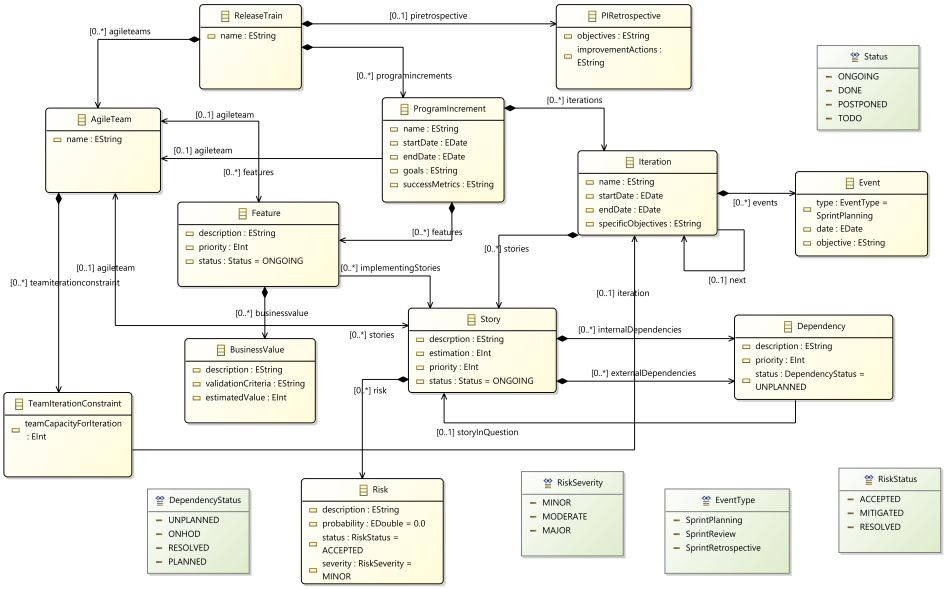

Figure 2: A Metamodel for PI Planning (implemented using Eclipse EMF)

Through the creation of a Domain-Specific Model (DSL) for PI Planning, the metamodel proposed in Fig. 2 intentionally eliminates the possibility of unplanned structural relationships between instances, effectively guarding against human errors. Furthermore, additional constraints can be incorporated into the model using OCL to provide an extra layer of prevention against context sensitive mistakes [18]. For instance, the following OCL constraints can serve as illustrations. They illustrate how OCL can enhance clarity regarding what is anticipated within an internal dependency relationship, distinct from an external dependency relationship in the context of a specific story. In essence, an external dependency with respect to a particular story pertains to a dependency linked to a story encompassed by a different agile team's scope.

Algorithm V.1 A few examples of context sensitive constraints in the model

```
1  context Story
2    inv: self.internalDependencies.storyInQuestion->forAll(s| s.agileteam = self.agileteam)
3
4  context Story
5    inv: self.externalDependencies.storyInQuestion->forAll(s| s.agileteam <> self.agileteam)
```

Let us consider another sample OCL example to demonstrate how powerful can a tool backed by a formal model be with regard to classic collaborative tools.

Algorithm V.2 Team charge coherence check.

```
1  context TeamIterationConstraint
2  inv: self.teamCapacityForIteration >= self.iteration.stories.estimation->sum()
```

This OCL constraint ensures that no instantiated model will be valid if the capacity of a specific agile team for a given iteration is less than the cumulative estimations of the stories assigned to that team for that iteration.

- Traditional PI Planning tools do not include mechanisms to deter team members from allocating stories with a cumulative estimation exceeding their capacity. As a result, planning team members must exercise caution and repeatedly assess the alignment of their plan with capacity requirements manually. Given the dynamic nature of the planning process, where the planning board frequently undergoes changes, this verification can become an unnecessary burden. Using a formal model, this cumulative estimation can be automatically calculated by the model, and the diagramming board can incorporate a signaling mechanism to indicate misalignment if it occurs.

The core metamodel depicted in Fig 2 is specifically tailored to the PI Planning activity, as shown. Consequently, our analysis is confined to the concepts and relationships within this core metamodel. However, by augmenting the core metamodel with an extension that includes additional domains, we have the capability to broaden our analyses to encompass those areas.

5.2.1 Value Stream Extension for Strategic Alignment

While the core metamodel focuses on the operational aspects of PI Planning (teams, features, dependencies, risks), it does not by itself capture the strategic dimension—specifically how planned work aligns with business value and organizational objectives. To address this limitation, we propose an extension of the metamodel incorporating elements of business value streams.

A *Value Stream* in Agile at scale represents a series of steps that deliver value to customers. It structures work around the flow of value, rather than traditional departments or hierarchies. By linking PI artifacts such as Features or Epics to specific value streams and strategic business goals, organizations gain the ability to assess alignment, prioritize more effectively, and make trade-offs visible during planning.

This extension allows planners and stakeholders to visualize the distribution of planned work across value streams, assess the coverage of business objectives, and identify potential misalignments. For example, during a PI Planning event, it becomes possible to detect if no planned feature contributes to a key business goal—a signal that strategic intent may not be effectively operationalized.

Furthermore, automated checks can be defined using OCL to require for instance that every business goal is covered by at least one feature in the planning. This constraint ensures that each strategic objective is addressed by at least one planned Feature, helping teams and business owners validate strategic coverage during the planning phase.

The value stream extension is a critical step in elevating PI Planning from team-level coordination to strategic portfolio alignment.

The extension brings new benefits including the following:

- **Strategic Alignment**: This analysis ensures that the value streams and features are in sync with the organization's new goals. Aligning work with strategic objectives is crucial for maintaining relevance and ensuring that efforts contribute directly to the overarching mission and vision of the organization.

- **Resource Optimization**: By identifying and focusing on the value streams and features that align with new goals, resources can be allocated more efficiently. This means prioritizing work that adds the most value and deprioritizing or eliminating work that is less aligned with current objectives.
- **Adaptability and Responsiveness**: The business environment is dynamic, and goals can change due to various factors like market trends, customer needs, or technological advancements. This analysis allows for agility in adapting plans to these changes, ensuring the organization remains responsive and competitive.
- **Risk Mitigation**: Misalignment between planned initiatives and organizational goals can lead to wasted resources, lost opportunities, and even reputational damage. By regularly aligning PI Plans with current goals, these risks are minimized.
- **Stakeholder Satisfaction**: Ensuring that value streams and features align with new goals can improve stakeholder satisfaction, as it demonstrates a commitment to delivering value that is relevant and timely. This is especially important for external stakeholders like customers and investors
- **Continuous Improvement**: This kind of analysis fosters a culture of continuous improvement. It encourages regular re-evaluation of plans and strategies to ensure they are effective and aligned with the organization's evolving goals.
- **Clarity and Communication**: Performing this analysis and adjusting plans accordingly provides clarity to teams about what is expected and why changes are being made. This enhances internal communication and can improve team morale and engagement.
- **End-to-End Visibility**: By incorporating the Value Stream concept into the metamodel, organizations gain a holistic view of the end-to-end process of delivering value. This allows stakeholders to understand the entire journey from ideation to delivery and beyond, facilitating better decision-making and resource allocation.
- **Alignment with Business Goals**: The Value Stream extension allows organizations to align their value delivery efforts directly with strategic business goals. By associating Value Streams with Business Goals, stakeholders can ensure that the work being done contributes directly to the organization's objectives.
- **Data-Driven Decision Making**: With Value Stream Metrics integrated into the metamodel, organizations can make data-driven decisions about where to focus improvement efforts. By analyzing performance data and trends, stakeholders can identify patterns, root causes, and opportunities for optimization.

The metamodel extension is illustrated in Fig 3. It's worth noting that various extensions can be applied to the core metamodel. These modifications enable it to cover additional related areas, thus allowing for analyses that span multiple domains.

VI A MODEL-DRIVEN FRAMEWORK FOR STAKEHOLDER-CENTRIC PI PLANNING

Program Increment (PI) planning involves orchestrating work items for multiple teams with the ultimate goal of achieving a shared outcome. The planning process generates various representations to address specific areas of concern. For instance, one representation may emphasize the dependencies between work items to highlight the chosen strategy for managing them, while another representation might concentrate on the identified risks to underscore the mitigation measures taken to address them.Moreover, since various user profiles participate in the process and may exhibit varying preferences for different representations, the concepts and capabilities of viewpoints and views offered by Model-Driven Engineering (MDE) align well with this

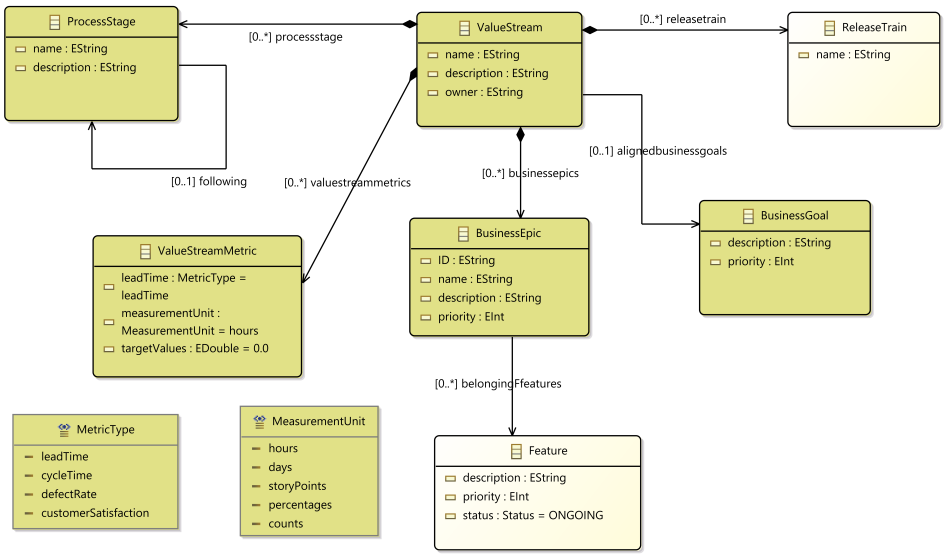

Figure 3: A metamodel extension to enable PI activities alignment with business value streams

diversity.A tool such as Eclipse Sirius is appropriate for implementing diverse viewpoints and views to cater to the distinct requirements of planning stakeholders as represented in Fig 4. Each viewpoint corresponds to a defined collection of representations, which may include diagrams, tables, matrices, or trees. These representations can be customized and expanded as needed. To achieve this, Eclipse provides model handling capabilities with EMF as well as graphical editing capabilities with GMF [19].

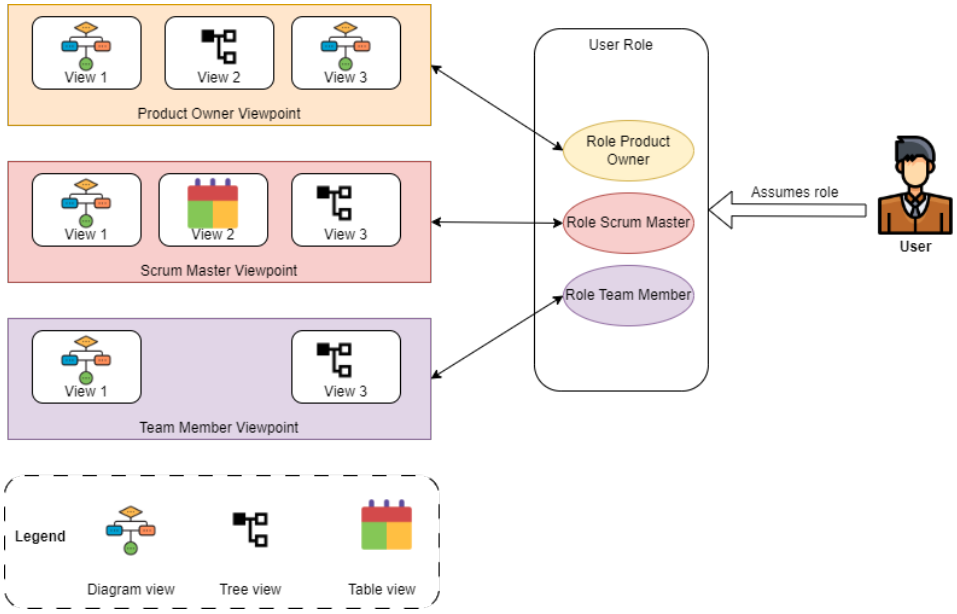

Figure 4: An illustration of user interaction with Viewpoints and Views

To make it concrete let us demonstrate how well-designed and thought viewpoints and views can enhance efficiency in PI management with appropriate traceability and impact analyses. This is made possible because of the capability available for model querying using OCL for example [20]. Table 2 summarizes PI concerns that can be efficiently handled without human effort. This is possible because these representations are backed by a formal model that allows transitive computations.

Table 2: Overview of scenarios where traceability and impact analyses can be managed using views supported by formal models

Type of problem	Problem or case description	Benefit
Feature to User Story Traceability	Ensure that each Feature is traceable to one or more User Stories.	Helps ensure that Features are broken down into actionable User Stories, facilitating development and testing.
Dependency Tracking	Track and trace Dependencies between Features, User Stories, or Teams.	Helps in identifying critical dependencies and their impact on planning, ensuring that dependencies are managed effectively.
Risk Management	Trace Risks to Features, User Stories, or Teams and monitor their mitigation progress	Enables proactive risk management and ensures that potential disruptions are addressed
Team Capacity and Commitment	Ensure that User Stories are assigned to Teams, considering their capacity and commitment	Facilitates balanced work distribution among Teams and supports commitment-based planning
Change impact analysis	A change request is received for a specific User Story	Determine the impact of the change on related Features, other User Stories, and the overall PI Plan
Resource Availability Change Analysis	One of the Teams experiences a change in resource availability.	Assess how the change affects the Team's capacity and the distribution of work items
Dependency Resolution Analysis	A critical dependency is resolved ahead of schedule	Evaluate the positive impact on Features, User Stories, and Teams, potentially accelerating delivery
Value Stream Alignment Analysis	A new strategic goal is defined at the organizational level	Determine which Value Streams and Features are aligned with the new goal, and adjust PI Plans accordingly

Let us take, for instance, the entry in Table 2, which pertains to Dependency Tracking. Consider calculating, for a given agile team within the Program Increment (PI), how it relies on all other teams. Referring to the metamodel, it's evident that dependencies between distinct agile teams are not explicitly stated and must be inferred from the stories the teams are involved in, as well as the dependencies that impact those stories. The subsequent OCL expression illustrates how this collection can be computed to provide data for a specialized view presentation.

Algorithm VI.1 Teams a given team depends on.

```
1  context AgileTeam
2  def teamsWeDependOn = self.stories.externalDependencies.storyInQuestion.agileteam ->asSet()
```

Let us now examine the entry in Table 2 regarding Value Stream Alignment Analysis. It's important to recognize that conducting this analysis isn't feasible with the core metamodel due to its lack of essential elements. However, the metamodel extension enables this analysis in addition to several other potential analyses not outlined in the table. The introduction of this new capability through the metamodel extension illustrates how seamlessly our approach, backed by formal modeling, facilitates the straightforward implementation of additional analyses. These analyses aid in positioning the PI activities within a broader strategic context aligned with the enterprise vision.

VII DISCUSSION

7.1 Theoretical and Practical Implications

This study demonstrates how MDE can enhance PI Planning by introducing formality, consistency, and automation. Our proposal lays a theoretical foundation for future DSL-based planning tools, while offering practical benefits such as better coordination, reduced error risk, and improved visibility.

7.2 Comparison with Existing Approaches

Unlike informal digital boards or backlog tools, our approach introduces structural constraints, cross-view traceability, and customizable viewpoints. This formalization enables systematic analyses that are difficult or impossible to perform manually. To further clarify the added value of our approach, we compare it in Table 3 with two widely used tools that support PI Planning events in practice: visual collaboration platforms (e.g., Mural, Klaxoon) and backlog management systems such as Jira Advanced Roadmaps. While these tools provide flexible ways to organize work and visualize plans, they lack formal semantics, traceability, and analytical capabilities. In contrast, our metamodel-based approach enables formal validation, customizable stakeholder views, and strategic alignment analysis through MDE techniques.

Table 3: Comparison with Existing PI Planning Tools and Approaches

Criteria	**Mural / Klaxoon**	**Jira Advanced Roadmaps**	**Our Metamodel-based Approach**
Formal structure	Informal board	Partially structured	Fully formal (Ecore + OCL)
Traceability between artifacts	Manual	Basic links (e.g. epics/stories)	Automated and bidirectional
Strategic alignment (value streams)	Not supported	Limited support	Explicit modeling and analysis
Custom stakeholder views	Static view	Role-based filters only	Multi-view modeling (RTE, PO, BO, etc.)
Impact analysis (what-if, changes)	Not available	Limited to issue links	Model-based reasoning with OCL
Constraint checking (e.g., capacity)	Manual validation	Partial rules	Formal OCL constraints
Extensibility	None	Limited plugins	Metamodel extensible via MDE tools

7.3 Limitations and Validity Threats

Construct validity. The metamodel proposed in this study was informed by informal observations and participation in multiple PI Planning events within a large banking organization. While this ensures relevance to real-world practices, the absence of a formal data collection protocol (e.g., structured interviews or surveys) may introduce interpretation bias. Nevertheless,

the iterative refinement of the model and validation through constraint formalization (OCL) help ensure that key concepts, roles, and relationships in PI Planning are accurately captured.

External validity. The current metamodel and framework were primarily shaped by observations in a specific domain—large-scale software development in the financial sector. While the structural elements of PI Planning (e.g., teams, features, dependencies, value streams) are broadly applicable across Agile Release Trains (ARTs), further studies are needed to assess the model's generalizability to other industries (e.g., telecom, automotive) and Agile-at-scale frameworks beyond SAFe. Future iterations of the model may benefit from cross-sector validation or adaptation guidelines.

Future empirical evaluation. As this work is primarily conceptual and model-driven, no large-scale empirical study has been conducted yet. The next stage of this research involves applying the proposed framework in a real-world PI Planning environment. This may take the form of an action-research collaboration with Agile teams, where the metamodel and viewpoint tools are deployed in practice. Evaluation metrics will include modeling effort, planning consistency, stakeholder satisfaction, and alignment quality. A comparative study with traditional planning tools (e.g., Jira + Mural) may also be conducted to assess effectiveness and usability.

7.4 DSL Implementation Roadmap

This study represents the first stage in a structured roadmap toward the implementation of a full Domain-Specific Language (DSL) for Program Increment (PI) Planning. The process is organized into four main steps:

1. **Metamodel Definition (Completed):** A core ecore-based metamodel was defined to capture the structural semantics of PI Planning events, including teams, features, stories, risks, and value streams.
2. **Viewpoint Modeling (In Progress):** Using Sirius, we are developing customized graphical viewpoints to support the distinct needs of different stakeholders (e.g., Release Train Engineers, Product Owners, Business Owners). These views are derived directly from the metamodel.
3. **Model Transformations and Constraints:** Transformation rules (e.g., validation, dependency propagation) and OCL-based constraints will be developed to support real-time feedback and automated reasoning during planning.
4. **Code Generation and Integration (Future Work):** The final step will involve integrating the DSL with agile project management tools (e.g., Jira, GitLab) via model-to-text transformations or API connectors, enabling end-to-end support from planning to execution.

This staged roadmap clarifies the progressive construction of the DSL and highlights how the current work lays a formal and conceptual foundation for future automation, validation, and tool support.

VIII CONCLUSION AND FUTURE WORKS

This paper introduced a metamodel and a model-driven framework to support formal modeling and analysis of Program Increment (PI) Planning events within large-scale Agile software development contexts. By capturing key planning artifacts, relationships, and constraints through an ecore-based metamodel enriched with OCL rules, our approach supports consistency checking, traceability, and alignment with strategic value streams—features not supported by existing PI Planning tools.

The proposed framework supports stakeholder-specific views, enabling Release Train Engineers, Product Owners, and Business Owners to reason about capacity constraints, dependencies, and strategic coverage from different perspectives. An extension of the core metamodel was introduced to explicitly model value streams and strategic objectives, making it possible to analyze business alignment during the planning phase.

Compared to popular tools such as Mural or Jira Advanced Roadmaps, our contribution brings formality, automation, and extensibility through the use of Model-Driven Engineering (MDE) techniques.

To our knowledge, this is the first metamodel and model-driven framework tailored to PI Planning that explicitly models both operational and strategic alignment artifacts, offering automated reasoning capabilities beyond what current Agile tools provide.

Nevertheless, this study is conceptual and has not yet been validated in a real-world industrial setting. The current prototype lacks implementation as a fully functional tool, and collaboration workflows are not yet modeled. These are important areas for further development.

Future research will focus on several directions:

- Implementing the metamodel and views as a prototype tool using Eclipse Sirius;
- Validating the approach through industrial case studies or simulated PI Planning sessions;
- Extending the metamodel to support stakeholder collaboration workflows and decision-making logic;
- Integrating DevOps metrics and risk analysis models to enhance traceability from planning to delivery.

In general, this work contributes to bridging the gap between Agile planning practice and the formal modeling capabilities offered by MDE. It also contributes to the foundational exploration of how formal methods and MDE can enhance coordination, traceability, and strategic alignment in scaled Agile environments. The proposed metamodel and framework can inspire similar modeling approaches for other Agile events such as Inspect & Adapt and System Demos.

REFERENCES

Publications

[1] C. Silva and A. Goldman. "Agile Methods Adoption on Software Development -a Pilot Review". In: *Proceedings of the 2014 Agile Conference*. Kissimmee, FL, USA, July 2014.

[2] M. Neumann, E.-M. Schön, M. Senapathi, M. Rauschenberger, and T. S. da Silva. *Towards a Closer Collaboration Between Practice and Research in Agile Software Development Workshop: A Summary and Research Agenda*. 2025. arXiv: 2507.10785 [cs.SE].

[3] Á. Csiszárik-Kocsir, H. Parameswaran, Á. Mészáros, and I. M. Tóth. "Exploring identification with the agile approach through an intercultural lens1". In: *2025 IEEE 29th International Conference on Intelligent Engineering Systems (INES)*. 2025, pages 000399–000404.

[4] R. Ghahremaninejad, K. Par, A. U. Peker, E. Can, and D. D. Acar. "Practicing Tailored Agile Framework for Automated Vehicle Software Development". In: *2025 IEEE 3rd International Conference on Mobility, Operations, Services and Technologies (MOST)*. 2025, pages 223–228.

[5] K. L. Oftebro, A. Nguyen-Duc, and K.-K. Kemell. *GenAI-Enabled Backlog Grooming in Agile Software Projects: An Empirical Study*. 2025. arXiv: 2507.10753 [cs.SE].

[6] B. Estácio, R. Prikladnicki, M. Morá, G. Notari, P. Caroli, and A. Olchik. "Software Kaizen: Using Agile to Form High-Perfomance Software Development Teams". In: *Proceedings of the 2014 Agile Conference*. Kissimmee, FL, USA, July 2014.

[7] P. Jain, A. Sharma, and L. Ahuja. "The Impact of Agile Software Development Process on the Quality of Software Product". In: *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*. Noida, India, Aug. 2018.

[8] M. Singh, N. Chauhan, and R. Popli. "A Framework For Transitioning Of Traditional Software Development Method To Distributed Agile Software Development". In: *2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*. Ghaziabad, India, Sept. 2019.

[9] A. Avasthi and G. Mishra. "A New Framework for the Agile Software Development Method". In: *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*. Coimbatore, India, Mar. 2018.

[10] J. de Lara, E. Guerra, and J. S. Cuadrado. "A posteriori typing for model-driven engineering". In: *2015 ACM/IEEE 18th International Conference on Model Driven Engineering Languages and Systems (MODELS)*. Ottawa, ON, Canada, Sept. 2015.

[11] M. Dalibor, N. Jansen, B. Rumpe, L. Wachtmeister, and A. Wortmann. "Model-Driven Systems Engineering for Virtual Product Design". In: *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. Munich, Germany, Sept. 2019.

[12] H. Alfraihi and K. Lano. "Trends and Insights into the Use of Model-Driven Engineering: A Survey". In: *2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. 2023, pages 286–295.

[13] B. Trask and A. Roman. "Leveraging Model Driven Engineering in Software Product Line Architectures". In: *2011 15th International Software Product Line Conference*. Williamsburg, VA, USA, Aug. 2011.

[14] A. Jossic, M. D. Del Fabro, J. Lerat, J. Bezivin, and F. Jouault. “Model Integration with Model Weaving: a Case Study in System Architecture”. In: *2007 International Conference on Systems Engineering and Modeling*. Chicago, IL, USA, Mar. 2007.

[15] M. L. Alvarez, I. Sarachaga, A. Burgos, E. Estévez, and M. Marcos. “A Methodological Approach to Model-Driven Design and Development of Automation Systems”. In: *IEEE Transactions on Automation Science and Engineering* 15.1 (2018), pages 67–79.

[16] N. Sannier and B. Baudry. “Toward multilevel textual requirements traceability using model-driven engineering and information retrieval”. In: *2012 Second IEEE International Workshop on Model-Driven Requirements Engineering (MoDRE)*. Chicago, IL, USA, Sept. 2012.

[17] C. Ebert and M. Paasivaara. “Scaling Agile”. In: *IEEE Software* 34.6 (2017), pages 98–103.

[18] K. Hassam, S. Sadou, V. Le Gloahec, and R. Fleurquin. “Assistance System for OCL Constraints Adaptation during Metamodel Evolution”. In: *2011 15th European Conference on Software Maintenance and Reengineering*. Oldenburg, Germany, Mar. 2011.

[19] M. L. Bernardi, J. A. Di Lucca, and D. Distante. “A model-driven approach for the fast prototyping of web applications”. In: *2011 13th IEEE International Symposium on Web Systems Evolution (WSE)*. Williamsburg, VA, USA, Sept. 2021.

[20] L. Iovino, A. Rutle, A. Pierantonio, and J. Di Rocco. “Query-Based Impact Analysis of Metamodel Evolutions”. In: *2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. Chicago, IL, USA, Aug. 2019.

A ACKNOWLEDGEMENTS

A previous version of this paper has been presented during the JRI 2023 Conference.