Towards a Perspective-Based Multi-Agent Architecture for Requirements Engineering of ML-Enabled Systems

Jose M C Boaro
Informatics Department, Pontifical Catholic University of Rio de Janeiro
Rio de Janeiro, Brazil
jboaro@inf.puc-rio.br

Matheus P. Soranço de Carvalho
Informatics Department, Pontifical Catholic University of Rio de Janeiro
Rio de Janeiro, Brazil
matheussoranco@gmail.com

Marcos Kalinowski
Pontifical Catholic University of Rio de Janeiro (PUC-Rio)
Rio de Janeiro, Rio de Janeiro, Brazil
kalinowski@inf.puc-rio.br

Abstract

[Context] Requirements Engineering (RE) for Machine Learning (ML)-enabled systems requires reconciling multiple, interdependent concerns, spanning system objectives, user experience, infrastructure, model, and data, that are difficult to capture through a single, monolithic specification process. Large Language Models (LLMs) show promising results in supporting RE tasks; however, when applied as general-purpose models to specify ML-enabled systems, they tend to produce shallow, monolithic specifications that ignore the dependencies among these concerns, while raising cost, privacy, and sustainability issues. **[Goal]** In this vision paper, we aim to contribute to AI-assisted requirements engineering of ML-enabled systems by discussing the feasibility of using Small Language Models (SLMs) to support this task. **[Method]** We ground our vision in PerSpecML, a perspective-based ML specification approach validated in academia and industry, whose five perspectives define the scope and boundaries of specialized agents. **[Results]** Our vision comprises a multi-agent architecture in which each agent is instantiated with an SLM fine-tuned to its corresponding perspective and coordinated by an orchestrator agent, under sequential, parallel, or hybrid topologies, with chain-of-thought reasoning intended to support reviewable rationale and human-in-the-loop review. **[Conclusion]** We acknowledge that this proposal still demands empirical evaluation, and we present it as a starting point for the community to discuss the feasibility of perspective-based multi-agent RE.

Keywords

Requirements Engineering, Machine Learning-Enabled Systems, Multi-Agent Systems, Small Language Models, PerSpecML

1 Introduction

The increasing use of Machine Learning (ML) models as core components in software systems has highlighted the challenges of engineering such systems, urging for adequate requirements engineering practices that account for the specificities of ML-based software. Specifying such systems requires reconciling multiple, interdependent concerns, spanning system objectives, user experience (UX), infrastructure, model, and data, that are difficult to capture through a single, monolithic specification process. In practice, specifying these concerns remains a manual, labor-intensive process that depends on coordinating multiple specialists.

Large Language Models (LLMs) are becoming increasingly ubiquitous in software engineering activities such as testing and code generation. In RE specifically, LLMs have been applied to requirements elicitation, analysis, and specification, showing promising results as document agents for extracting domain models and reducing conflicts and ambiguity [14–16]. However, when applied as a single, general-purpose model to specify ML-enabled systems, LLMs face known limitations such as requirements overload and limited domain-specific knowledge [4], which may hinder their ability to capture the dependency structure among the interdependent concerns of such systems. Beyond this limitation, their use carries significant downsides: cost, as the most capable models are commercialized as proprietary products; privacy, as input data is processed and stored in third-party infrastructure; and carbon footprint, as these models require substantial energy to operate and train. Furthermore, since LLMs are trained on broad-scope data, achieving domain specificity through prompt-based approaches alone remains a challenge.

This vision paper discusses the feasibility of applying Small Language Models (SLMs) to overcome the limitations outlined above, as they require fewer computational resources to be operationalized and trained, thus addressing privacy and energy consumption concerns. To ground the application of SLMs for requirements engineering, we propose investigating the use of PerSpecML [11], a requirements engineering approach for ML-enabled systems validated in academia and industry, as the training artifact for the SLMs. Specifically, we propose a perspective-based multi-agent architecture for AI-assisted requirements engineering of ML-enabled systems, using PerSpecML's five perspectives to define the scope and boundaries of specialized agents, where each agent is instantiated with an SLM fine-tuned to its corresponding perspective, addressing inter-perspective consistency and dependency challenges. This proposal demands evaluation, and this paper is an opening for the community to discuss the feasibility of this approach.

The remainder of this paper is structured as follows: Section 2 presents the background on RE for ML-enabled systems, PerSpecML, and agentic systems in SE. Section 3 describes the proposed perspective-based multi-agent architecture. Section 4 discusses open challenges. Section 5 concludes and outlines future work.

2 Background

2.1 Requirements Engineering for ML-Enabled Systems

ML-enabled systems are software systems in which one or more components rely on ML models to deliver part of the system's functionality [8]. Because the behavior of ML components emerges from data rather than explicit specification, it is harder to anticipate, specify, and verify, introducing concerns such as fairness, explainability, iterative experimentation, and implicit assumptions about data and model performance [5, 12].

Empirical evidence shows that RE remains one of the most critical and least mature activities in ML projects. An international survey with 188 practitioners reported that RE activities are often conducted by project leaders and data scientists rather than requirements engineers, that requirements are documented informally, and that unclear goals, limited domain understanding, and difficulty aligning requirements with available data are among the most pressing problems [1, 8]. Studies on non-functional requirements (NFRs) for ML-enabled systems further show that properties such as data quality and explainability remain poorly operationalized [5]. These findings motivate dedicated RE approaches for the multifaceted, multi-stakeholder nature of ML-enabled systems.

2.2 PerSpecML

PerSpecML is a perspective-based approach for specifying ML-enabled systems, helping practitioners identify which ML and non-ML attributes are relevant to overall system quality [10, 11]. Its central premise is that the concerns of an ML-enabled system are too heterogeneous for a monolithic specification and should be organized into five perspectives: system objectives, user experience, infrastructure, model, and data. Each perspective groups a catalog of concerns to be analyzed collaboratively with stakeholders best positioned to reason about them, supported by a task-and-concern diagram and a specification template [11]. PerSpecML was evaluated in academia and in two industrial case studies, where it proved useful in revealing important requirements that would otherwise have been missed [11]. The modularity of its perspectives (each with well-defined boundaries, dedicated stakeholders, and distinct quality criteria) makes PerSpecML a natural foundation for decomposing the RE process for ML-enabled systems, which we leverage to define the boundaries of specialized agents.

2.3 Agentic Systems in Software Engineering

LLM-based agents couple a language model with a role profile, context perception, actions (e.g., tool invocation), and mechanisms for interacting with other agents [3]. LLM-based multi-agent systems decompose complex problems among specialized agents coordinated by an orchestrator, improving robustness and scalability [3, 6], and have been applied across the software engineering lifecycle, increasingly including requirements elicitation and analysis [6].

Two techniques are particularly relevant to agentic RE. Chain-of-thought prompting elicits intermediate reasoning steps, improving multi-step reasoning and exposing the rationale behind each decision to support human-in-the-loop review [13]. In addition, Small Language Models (SLMs) are well suited for the narrow, format-constrained subtasks that agents perform, rivaling larger models while being cheaper to serve, easier to fine-tune, deployable locally, and aligned with Green AI principles and the privacy of sensitive project information [2, 9]. These properties make SLM-based multi-agent architectures a promising substrate for perspective-oriented RE.

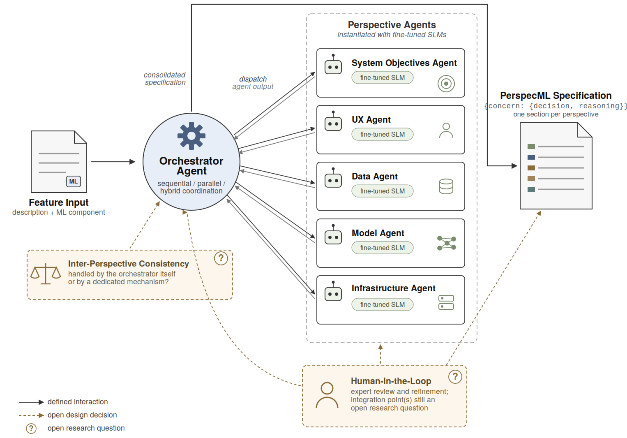

Figure 1: Overview of the proposed architecture, common to all coordination topologies (Section 3). Dashed elements denote open design decisions: inter-perspective consistency and human-in-the-loop integration.

3 Perspective-Based Multi-Agent Architecture for RE

3.1 Motivation

Generalist language models tend to produce shallow specifications for ML-enabled systems. Our preliminary observations with small language models confirm this: in zero-shot and few-shot prompting, generated outputs showed minimal explicit reasoning over perspective-specific trade-offs, with chain-of-thought rates below 3%, compared to over 37% after per-perspective fine-tuning. This reflects an inherent dependency structure among ML requirements perspectives: infrastructure concerns, for instance, can only be defined after the model is decided, so not all concerns can be specified independently within a single context prompt. In real ML projects, RE is typically conducted by a team of specialists coordinated by a requirements engineer.

Figure 1 shows an overview of the proposed architecture, which replicates the previously described organization as an agentic solution. Each agent assumes the role of a domain specialist grounded in a PerSpecML perspective, coordinated by an Orchestrator Agent. The modularity of PerSpecML's five perspectives, each with well-defined boundaries, dedicated stakeholders, and distinct quality criteria, ensures that each agent operates within a clearly scoped domain with concrete evaluation standards for its outputs.

3.2 Architecture Overview

The proposed multi-agent architecture comprises six agents: one orchestrator and five perspective agents (System Objectives, UX, Data, Model, and Infrastructure). The Orchestrator Agent distributes the system feature description to each perspective agent and coordinates the resolution of inter-perspective inconsistencies. Each perspective agent then receives the feature context and specifies the artifacts related to its corresponding PerSpecML perspective. The execution flow across agents is determined by the dependency structure among perspectives.

We propose three interaction topologies. In a sequential topology, perspectives with direct dependencies are elicited in order (e.g., Data before Model, since model selection criteria depend on data availability and quality constraints). In a parallel topology, independent perspectives are elicited simultaneously, reducing overall execution time. In practice, a hybrid topology is the most likely configuration, combining sequential execution among dependent perspectives with parallel execution among independent ones.

Consider a feature description such as "real-time fraud detection for credit card transactions." Under the hybrid topology, the Data Agent first specifies data-related concerns (e.g., data source, train/test split, class imbalance), since these constrain what is feasible downstream. The Model Agent then defines model-related concerns (e.g., latency requirements ruling out certain architectures) conditioned on the Data Agent's output, while the UX and Infrastructure Agents operate in parallel, as their concerns are largely independent of the Data-Model chain. If the Infrastructure Agent's output assumes a deployment latency incompatible with the Model Agent's chosen approach, this conflict is flagged as an inter-perspective inconsistency for Orchestrator-mediated resolution, potentially requiring human input (Section 4.3).

As an alternative, a peer-to-peer topology allows the five perspective agents to freely exchange information and request clarifications, with the Orchestrator acting as mediator rather than central coordinator; this is suited for scenarios where inter-perspective dependencies emerge dynamically during elicitation.

We adopt one agent per PerSpecML perspective as the default decomposition granularity (Section 2), though concerns are not always cleanly separable in practice: a data scientist and a data engineer, for instance, routinely collaborate on overlapping concerns rather than working in isolation. The peer-to-peer topology reflects this reality, allowing agents to jointly resolve overlapping concerns rather than requiring strict non-overlap upfront. Whether this scales to concerns spanning three or more perspectives, or whether some perspectives require decomposition into sub-agents, is a design question beyond this work.

Each perspective agent generates its artifacts using chain-of-thought reasoning to capture the rationale behind each specification decision. Beyond producing PerSpecML artifacts, each agent explains the reasoning behind its choices (scope decisions, trade-offs, assumptions about system context), supporting review by both human stakeholders and the Orchestrator Agent. Such rationale reflects the agent's stated reasoning rather than a verified trace of its internal decision process; grounding auditability in provenance records, tool calls, or assumption logs remains an open direction (Section 4).

3.3 Small Language Models as Perspective Agents

One of the main objectives of this work is to investigate the use of Small Language Models to specify ML-enabled system requirements. Unlike commercial LLMs, SLMs can be fine-tuned to specific domains using accessible computational infrastructure, directly addressing the lack of specificity that general-purpose models exhibit, a limitation particularly critical when architecting ML-enabled systems. We hypothesize that SLMs offer additional advantages when instantiating the perspective agents described: each agent requires less computational resources per inference call than a single large-scale LLM, though a multi-agent system composed of several SLMs may require more total inference operations overall, a trade-off to be empirically assessed. Our preliminary observations with fine-tuned SLMs are encouraging but mixed: chain-of-thought reasoning depth improved substantially after fine-tuning, but structural adherence to the PerSpecML format did not improve uniformly across models, indicating that gains may be model-dependent.

Beyond performance, SLMs' lower energy footprint and support for local deployment (Section 2) motivate our premise of instantiating each perspective agent with a fine-tuned SLM, keeping sensitive project information within the user's organization.

4 Discussion

4.1 Inter-Perspective Consistency

Individualized agents may produce decisions that conflict across perspectives, e.g., the Data Agent selecting training data whose disk space requirements exceed the Infrastructure Agent's memory and latency constraints. Each decision may be correct within its own domain, yet unresolved conflicts across perspectives directly impact the overall quality of the specification.

The Orchestrator Agent is responsible for addressing these conflicts, but exactly how it should act remains an open question. One alternative is to signal a human reviewer when conflicts arise, delegating the decision to the requirements engineer and then verifying that the resolved requirements are compliant with overall user needs. Alternatively, the Orchestrator Agent could proactively resolve conflicts, leveraging its global knowledge of all perspectives; this requires further investigation of prioritization criteria, particularly whether priority should be derived from the system context or defined explicitly by the human stakeholder. The communication mechanisms between agents also require investigation. For well-defined inter-perspective dependencies, agents can communicate deterministically based on predefined precedence, but this may fail when dependencies emerge dynamically. For such cases, peer-to-peer communication is worth investigating: agents could request clarifications or validate specifications against another agent's output, with a mediator to ensure convergence. While the Orchestrator Agent can fulfill this role, whether a human reviewer should serve as mediator for conflicts requiring domain judgment beyond the agents' scope remains an open question.

4.2 Evaluation of Agent-Generated Requirements

When evaluating agent-generated requirements, one of the main challenges is assessing the quality of the specifications produced. Metrics such as BERT-F1 and ROUGE-L only assess semantic similarity between two sets of generated requirements and cannot determine whether the generated specification is, in fact, aligned with stakeholder needs and useful to the development team. A requirement may be well written but incomplete and infeasible to implement. This is even more critical in perspective-based agent-generated requirements, where specifications must not only be correct within each perspective but also consistent with one another. Furthermore, vague specifications can yield high metric scores but may not be considered good specifications, as they lack the detailed definitions required for implementation.

We argue that evaluating agent-generated requirements requires combining multiple complementary methods rather than relying on a single quality framework. ISO/IEC DIS 25059 [7], which extends ISO/IEC 25010 to cover AI/ML software specificities, offers a useful rubric for assessing whether specifications address functional correctness, robustness, fairness, explainability, transparency, reliability, and safety, mapped onto the corresponding perspectives (fairness and robustness for Model; reliability and safety for Infrastructure; explainability and transparency for UX; and functional correctness for System Objectives). However, it characterizes the qualities of an operating AI system rather than of a specification artifact, and does not by itself establish whether a generated requirement is complete, correct, or consistent across perspectives. We therefore complement it with RE-specific quality properties (completeness, correctness, inter-perspective consistency), assessed through structured expert review with inter-rater reliability measures such as Krippendorff's Alpha, and benchmarked against human-generated specifications and a single-agent monolithic baseline to determine whether perspective-based decomposition yields measurable gains.

4.3 Human-in-the-Loop

A perspective-based multi-agent system is not a final solution that can substitute human judgment: ultimately, a human must certify that the specified solution is adequate for the system. Our proposed solution serves as a tool to accelerate this process and increase software team productivity by reducing manual effort. Determining when and how the human integrates into this pipeline remains an open question [17]; in our view, the human can participate in the loop to: (a) provide initial context, (b) review the specification generated by the Orchestrator Agent, (c) be consulted to resolve conflicting specifications, or (d) review each perspective individually before consolidation by the Orchestrator Agent, being signaled based on the chain-of-thought generated by each agent.

The human role also differs from the traditional requirements engineer, who instead coordinates a team of agents rather than human specialists, accelerating elicitation and anticipating conflicts.

5 Conclusion and Future Work

Specifying requirements for ML-enabled systems demands multi-perspective reasoning and consistency verification across interdependent concerns, a challenge that monolithic language model approaches cannot adequately support. Our vision is a perspective-based multi-agent architecture using SLMs by design, where each agent specializes in one PerSpecML perspective and incorporates chain-of-thought reasoning to justify each decision. Under sequential, parallel, or hybrid topologies, this design treats inter-perspective consistency as a first-class property through agent collaboration, operating under data privacy, energy, and cost constraints while augmenting practitioners rather than replacing them. Beyond applying agentic systems to a core SE activity, our proposal engages challenges central to engineering agentic systems themselves: multi-agent coordination, human-in-the-loop integration, and systematic evaluation. We outline a falsifiable research agenda: (i) do perspective-specialized SLM agents produce more complete and consistent specifications than a single monolithic LLM baseline; (ii) which coordination topology (sequential, parallel, hybrid, or peer-to-peer) yields the best trade-off between specification quality and execution cost; (iii) how does the choice between fine-tuned SLMs and general-purpose LLMs affect the quality-cost balance; and (iv) under which conditions is human intervention necessary rather than optional. Progress requires empirical comparison against human-generated and single-agent baselines, using both automated metrics and structured expert evaluation, as discussed in Section 4.2.

Artifact Availability

This vision paper does not report empirical results.

Acknowledgements

We thank the Brazilian Research Council – CNPq (Grant 312275/2023-4), Rio de Janeiro State's Research Agency – FAPERJ (Grant E-26/204.256/2024), the Coordination for the Improvement of Higher Education Personnel (CAPES), and the Kunumi Institute for their generous support.

References

[1] Santos Alves, A. P., Kalinowski, M., Giray, G., Mendez, D., Lavesson, N., Azevedo, K., Villamizar, H., Escovedo, T., Lopes, H., and Gorschek, T. 2023. Status Quo and Problems of Requirements Engineering for Machine Learning: Results from an International Survey. In *PROFES 2023*. Springer, 159–177.

[2] Belcak, P., Heinrich, G., Diao, S., Fu, Y., Dong, X., Muralidharan, S., Lin, Y. C., and Molchanov, P. 2025. Small Language Models are the Future of Agentic AI. *arXiv preprint arXiv:2506.02153* (2025).

[3] Guo, T., Chen, X., Wang, Y., Chang, R., Pei, S., Chawla, N. V., Wiest, O., and Zhang, X. 2024. Large Language Model Based Multi-Agents: A Survey of Progress and Challenges. In *IJCAI 2024*. 8048–8057.

[4] Arora, C., Grundy, J., and Abdelrazek, M. 2024. Advancing Requirements Engineering Through Generative AI: Assessing the Role of LLMs. In *Generative AI for Effective Software Development*, Springer, 129–148.

[5] Habibullah, K. M., Gay, G., and Horkoff, J. 2023. Non-Functional Requirements for Machine Learning: Understanding Current Use and Challenges Among Practitioners. *Requirements Engineering* 28, 2 (2023), 283–316.

[6] He, J., Treude, C., and Lo, D. 2025. LLM-Based Multi-Agent Systems for Software Engineering: Literature Review, Vision, and the Road Ahead. *ACM TOSEM* 34, 5 (2025), 1–30.

[7] International Organization for Standardization. 2023. *ISO/IEC DIS 25059 – Quality Model for AI Systems*. ISO.

[8] Kalinowski, M., Mendez, D., Giray, G., Santos Alves, A. P., et al. 2025. Naming the Pain in Machine Learning-Enabled Systems Engineering. *Information and Software Technology* (2025).

[9] Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. 2020. Green AI. *Communications of the ACM* 63, 12 (2020), 54–63.

[10] Villamizar, H., Kalinowski, M., Lopes, H., and Mendez, D. 2022. Towards Perspective-Based Specification of Machine Learning-Enabled Systems. In *SEAA 2022*. IEEE, 112–115.

[11] Villamizar, H., Kalinowski, M., Lopes, H., and Mendez, D. 2024. Identifying Concerns When Specifying Machine Learning-Enabled Systems: A Perspective-Based Approach. *Journal of Systems and Software* 213 (2024), 112053.

[12] Vogelsang, A. and Borg, M. 2019. Requirements Engineering for Machine Learning: Perspectives from Data Scientists. In *REW 2019*. IEEE, 245–251.

[13] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., and Zhou, D. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *NeurIPS 2022*, Vol. 35. 24824–24837.

[14] Görer, B. and Aydemir, F. B. 2023. Generating Requirements Elicitation Interview Scripts with Large Language Models. In *REW 2023*. 44–51.

[15] Arulmohan, S., Meurs, M.-J., and Mosser, S. 2023. Extracting Domain Models from Textual Requirements in the Era of Large Language Models. In *MODELS-C 2023*. 580–587.

[16] Lutze, R. and Waldhör, K. 2024. Generating Specifications from Requirements Documents for Smart Devices Using Large Language Models (LLMs). In *HCII 2024*. Springer, 94–108.

[17] Xu, X., Zhang, D., Liu, Q., Lu, Q., and Zhu, L. 2025. Agentic RAG with Human-in-the-Retrieval. In *ICSA-C 2025*. 498–502.