A Method for Agent-led Requirements Discovery Towards System Specification

Peter Denno[1], Dohyeon Kong[2], Maya Reslan[1], Lorelai Lyons[1], Yinfeng Shen[3], Carolyn Psenka[3], Hyunbo Cho[2], and Kyoung-Yun Kim[3]

[1] *National Institute of Standards and Technology*
[2] *POSTECH*
[3] *Wayne State University*

Abstract

The predominant use case for large language model (LLM) AI is to receive answers to questions you pose; in this paper, we describe a method by which the AI poses the questions. Asking the right questions is essential to many engineering tasks including requirements engineering, system integration, and system evolution. The paper describes a method that uses orchestrated interviewer agents to formulate relevant questions and interpret responses for designing or refining an engineered system. The target applications for the method are human/AI teaming systems that mentor users in long-running and complex endeavors such as maintaining manufacturing digital twins and production scheduling systems. These interviews require deep domain expertise while remaining necessarily limited in scope due to practical constraints on system formulation and validation. The paper introduces *discovery schemas* and *tool stewards*. Discovery schema are templates that, in an interviewing process, shape the formulation of questions and the interpretation of responses to schema-conforming structures. Tool stewards are agentic wrappers that orchestrate such interviewing processes to adapt and integrate the wrapped tool for use in interviewees' system context. The paper demonstrates the approach through a working implementation with illustrative examples from manufacturing production scheduling.

1 Introduction

In the age of intelligent machines—with appropriate guidance—human system participants may become ideal system designers. First, however, there is a tension to overcome. Increasingly, automation replaces human roles in complex socio-technical systems. With the loss of human roles, understanding of system functions and architecture may correspondingly erode [1, 2]. System designers must anticipate the environment in which the system will operate, but as the actual operating environment strays from the anticipated one, system participants are apt to be the first to notice. When these participants are empowered with an understanding of system functions and architecture similar to that of system designers, they may be able to avert problems and exploit opportunities. This more encompassing system is thereby more resilient than one with restricted human roles. The concept of participants-as-designers described in the paper appears feasible in long-running endeavors such as manufacturing by small and medium-sized firms. The paper describes AI-assisted sensemaking essential to enabling the participants-as-designers paradigm. Sensemaking—the process by which people give meaning to their experiences and situations—transforms raw observations into coherent understanding that enables informed action [3]. In our context, sensemaking involves discussions of how the system works, the technical environment (data, interfaces, physical resources, etc.) in which it operates, and what is sought in the participants' envisaged program of improvement.

Participants-as-designers is a human/AI teaming form of participatory design [4] that assumes human participants can describe the high-level goals of their envisaged program to improve outcomes in the system. However, implementing that program may require expertise in myriad esoteric

technical areas; participants might not possess this expertise themselves. The human/AI teaming perspective asserts that needed expertise might come from any member, human or AI.

To realize this vision of empowered participants, we must address how relevant expertise is recognized and applied. With respect to recognizing the relevant expertise, the paper describes how incremental discovery of the participants' goals and the problem domain reveals a match to expert strategies. With respect to applying the expertise, the complete practice of systems engineering applies, from requirements elicitation to system validation. However, in nascent human/AI teaming scenarios it is additionally necessary to reiterate—for AI use—the understanding of system environment which the human participants already possess. In the case of teaming on production scheduling systems, for example, this includes established knowledge of the production process, production resources, information sources, and information used.

The sensemaking outlined above is achieved by orchestrated work with discovery schema in an agentic AI system [5]. *Discovery schemas* are schemas with integrated examples and annotations used to guide interviewer agents in the work of formulating questions and interpreting responses. The interviewer generates a *schema-conformed response* (SCR) by inspecting the discovery schema, formulating a question, and interpreting the interviewees' response according to the discovery schema's structure. Each SCR represents how the interviewer maps information from an interviewee response onto a facet of the discovery schema, generating new instances of the facet where necessary. The orchestrator reviews these SCRs to build or revise a structure of equivalent scope and form to the discover schema. Annotations in the discovery schema are associated with its properties and substructure, providing information about the value sought, the example response, and notes about the interpretation. The interviewer is encouraged to use the discovery schema's annotation syntax in its SCRs to note situations where the interviewees' response might merit follow-up, or where the schema cannot adequately capture the full intent of the interviewees' response. In some cases, the discovery schema describes a problem solving task rather than a simple schema; solving the problem requires information from the interviewees.

Discovery schema, the orchestrator's role in selecting them, and its role in integrating SCRs into a cohesive viewpoint are the principal concerns of the paper. Our exploratory implementation in production scheduling, however, applies this agentic interviewing methodology in a context where the goal is to integrate the use of a specific kind of computational tool, ones that can be operated through domain-specific languages (DSLs). In this context, the orchestrator and interviewers comprise an agentic system termed a *tool steward* that (1) wraps the DSL-enabled tool, mediating its configuration and integration, (2) mentors interviewees in use of the DSL, and (3) verifies accurate capture of interviewees' intent using diagrams built from aggregated SCRs. To address security concerns in industrial applications, tool stewards have no role in runtime of the DSL-enabled tool they configure and integrate.

Section 2 of the paper describes related work in interviewing and requirements elicitation. *Section 3* provides a detailed account of interviewer's use of discovery schema. This section includes discussion of elements of a interview conversation including (a) *warm-up questions* for surveying interviewees' concerns, viewpoint, and goals, and (b) *reframing questions* which assess whether interviewees believe the interview is capturing relevant concerns. *Section 4* describes the role of the interview orchestrator, an agent that determines the path that the interview takes by progressively selecting the most relevant next discovery schema to be used by the delegated interviewer. Choosing the path of the conversation is a task that relies heavily on domain expertise. Notes on the appropriate use of a discovery schema can be kept with the schema, but additional resources such as a knowledge graph of the relevant body of knowledge may be helpful. This section also discusses partitioning conversation into distinct areas of inquiry. Examples include tasks comprising a process, data used in the process, and resources available. Partitioning the conversation this way may make reviewing past discussions easier. *Section 5* concludes the paper with a discussion of limitations of the work and future plans.

2 Related Work

We start by considering work that also relies on a sensemaking discovery process, but goes about it differently and serves different needs: AI programming assistants.

3 Discovery Schema

3.1 Assumptions and Terminology

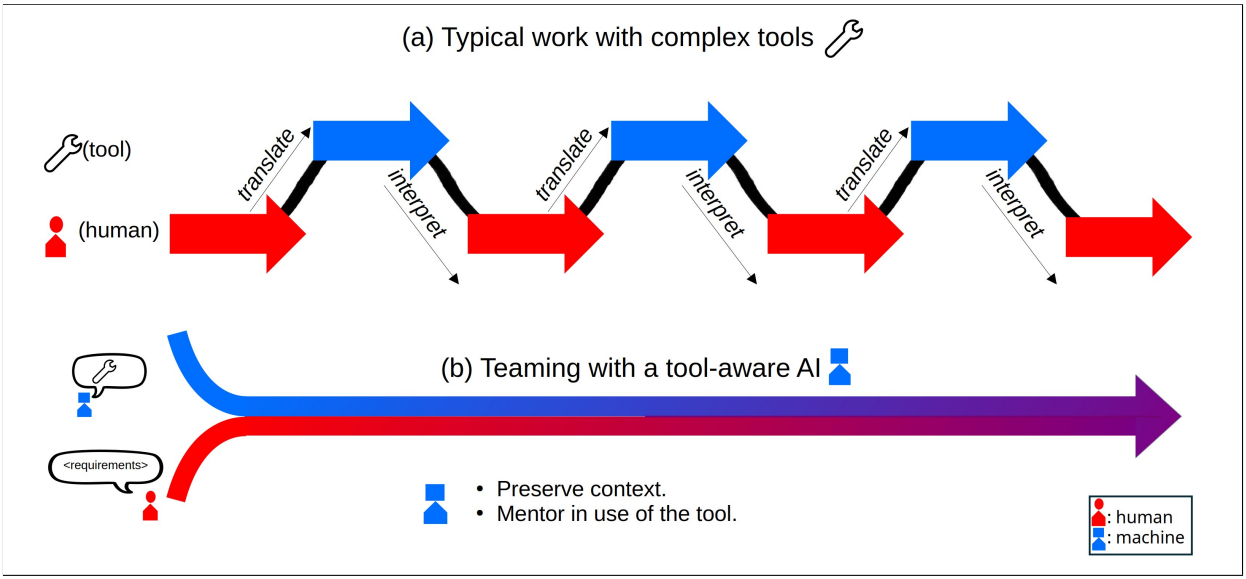
Human/AI teaming is human collaboration that additionally includes agentic AI as a participant. In the scope of the paper, projects where this collaboration is sought are assumed to (a) present cognitive (rather than physical) challenges primarily, (b) be long-running (weeks to years) and, (c) require solution to complex problems, including those with non-linear response, path dependence, coupling of effects, no decomposition to subproblems, and emergent behavior. It is often the case that software tools are used in the solution to complex problems. The responsibilities of the "AI participant" in the project are restricted to the team's use of such a tool; the AI participant's expertise is focused on use of the tool. Because human participants may not be familiar with the tool and its associated body of knowledge, we focus on tools that can be driven by a DSL, and further assume that the AI participant can mentor human participants in the use of the DSL. As collaboration proceeds, the contribution of the AI participant to the team's work is expressed in what is formulated in the DSL; it records solutions to requirements expressed by humans in the tool's domain-specific language. *Table* 1 provides examples of DSLs used with tools for various problem types.

Table 1: Example Tool DSLs and their usages.

DSL	**Problem Type**	**Technical body of knowledge**
MiniZinc	scheduling, resource allocation, supply chain optimization, portfolio selection	constraint programming, mathematical optimization, combinatorial search
Modelica	dynamic system behavior, physical system simulation, control system design	multi-domain physical modeling, differential-algebraic equation solving, continuous/hybrid simulation
Alloy	structural consistency, design validation configuration verification	relational logic modeling, constraint satisfaction, counterexample finding

The arrangement described above— in which a "tool-with-agency" is integrated into a team— is intended to improve the performance of the team by (1) automatically translating the participants' viewpoint into the tool's DSL, and (2) interpreting tool output back into that viewpoint. *Figure* 1 depicts workflows that use complex tools with and without human/AI teaming. As part (b) of the figure suggests, integrating the use of the tool begins with no assumptions about the intended role of the tool in the collaboration and develops through exposure to project's goals. In the context of the paper, *interviewing* is active pursuit and sensemaking of the project's goals; it is akin to requirements engineering in systems engineering processes. Provisions to verify and validate requirements expressed in interviews distinguishes our notion of agent-led interviewing from others, such as for writing autobiographies [6].

Figure 1: Part (a) of the figure depicts a common workflow in which decisions are made with the help of complex tools such as physics-based simulation or project planning software. Participants (red arrows) may meet to discuss their viewpoints and plans, leading to one or more participants translating a viewpoint into tool input, running the tool, and interpreting results for use by the team. Part (b) of the figure depicts the goal state, where, progressively, tool use is facilitated by reducing the effort required to translate the human viewpoint and interpret tool results. This reduction is owing to the agency of the machine component of the team, and its discovery of participant viewpoints through interviewing.



We call such a tool-with-agency a *tool steward*; it is an agentic system that wraps a tool to enable integration into target systems through requirements discovery and configuration. Specifically, the role of the tool steward is both (1) discover the environment in which the tool will operate (roughly speaking, answering the question "Where am I?" from the tool steward's point of view), (2) discover of the role which the tool will fill (roughly speaking, answering the question "What will you want me to do?" from that point of view). Interview questions about the environment may reveal, for example, that tool input will be obtained from spreadsheets uploaded to the tool. Alternatively, these questions might reveal that there are application protocol interfaces (APIs) to obtain the tool input. In either of these cases (and more) the goal is towards situating or embedding the tool in the extant technical environment. We distinguish tool stewards that have started an discovery process as *situated tool stewards*. In our work, by design, neither the tool steward nor situated tool steward play any role in the runtime use of the tool. In industrial settings, not adding an agentic system to the runtime may improve cybersecurity characteristics.

3.2 Expertise and Discovery Schema

4 Interview Organization and Orchestration

5 Conclusion

References

[1] Tapani Rinta-Kahila, The University of Queensland Business School, Esko Penttinen, Aalto University School of Business, Antti Salovaara, Aalto University / University of Helsinki, Wael Soliman, University of Agder, Norway / University of Jyväskylä, Joona Ruissalo, and Aalto University School of Business. The Vicious Circles of Skill Erosion: A Case Study of Cognitive Automation. *Journal of the Association for Information Systems*, 24(5):1378–1412, 2023.

[2] Hao-Ping (Hank) Lee, Advait Sarkar, Lev Tankelevitch, Ian Drosos, Sean Rintel, Richard Banks, and Nicholas Wilson. The Impact of Generative AI on Critical Thinking: Self-Reported Reductions in Cognitive Effort and Confidence Effects From a Survey of Knowledge Workers. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, pages 1–22, Yokohama Japan, April 2025. ACM.

[3] Karl E. Weick, Kathleen M Sutcliffe, and David Obstfeld. Organizing and the Process of Sensemaking. *Organization Science*, 16(4):409–421, March 2011.

[4] Rex Hartson and Partha S. Pyla. Design Production. In *The UX Book*, pages 333–357. Elsevier, 2012.

[5] Ranjan Sapkota, Konstantinos I. Roumeliotis, and Manoj Karkee. AI Agents vs. Agentic AI: A Conceptual Taxonomy, Applications and Challenges, May 2025.

[6] Jinhao Duan, Xinyu Zhao, Zhuoxuan Zhang, Eunhye Ko, Lily Boddy, Chenan Wang, Tianhao Li, Alexander Rasgon, Junyuan Hong, Min Kyung Lee, Chenxi Yuan, Qi Long, Ying Ding, Tianlong Chen, and Kaidi Xu. GUIDELLM: Exploring LLM-Guided Conversation with Applications in Autobiography Interviewing. In *Human Language Technologies (Volume 1: Long Papers)*, April 2025.