KiMO: Knowledge-infused Multi-agent Orchestrator

Demonstration Track

Ritvik Garimella
University of South Carolina
Columbia, United States
ritvikg@sc.edu

Michael Charles Stewart
University of South Carolina
Columbia, United States
mcs46@email.sc.edu

Kevin Do
University of South Carolina
Columbia, United States
kd79@email.sc.edu

Leeon Israel
University of South Carolina
Columbia, United States
lisrael@email.sc.edu

Amit Sheth
University of South Carolina
Columbia, United States
amit@sc.edu

ABSTRACT

Multi-Agent Systems (MAS) provide a natural paradigm for solving complex problems requiring coordinated, multi-step reasoning and decision making. As these systems scale, effective agentic orchestration becomes crucial. While existing tools such as AgentFlow and Agent Development Kit (ADK) support agent communication and task delegation, they remain limited in understanding task semantics, coordinating heterogeneous agent types (e.g., reactive vs. cognitive), and aligning agent selection with domain knowledge and infrastructure constraints. We present *KiMO*, a knowledge-infused multi-agent orchestrator that makes coordination explicit through structured knowledge infusion: *planning ontologies* encode domain-specific task decompositions, and *agent registries* formalize capabilities and compatibility constraints for heterogeneous components. KiMO operationalizes these through a two-stage pipeline: ontology-guided sub-task planning (PlanGen) and registry-guided workflow assembly (AgentGen). This dual knowledge infusion enables interpretable, flexible orchestration with human-inspectable coordination decisions. Our interactive platform demonstrates real-time workflow construction for manufacturing pipelines with expert refinement and dynamic knowledge extension. Code is available at https://tinyurl.com/3es9zyhj and Demo Video at https://tinyurl.com/yfa8ctwn.

KEYWORDS

Multi-agent orchestration; Workflow construction; Heterogeneity

ACM Reference Format:
Ritvik Garimella, Michael Charles Stewart, Kevin Do, Leeon Israel, and Amit Sheth. 2026. KiMO: Knowledge-infused Multi-agent Orchestrator: Demonstration Track. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 3 pages. https://doi.org/10.65109/WTIF5096

1 INTRODUCTION

Large Language Models (LLMs) have enabled automation beyond static rule execution, spurring interest in agent-driven workflow systems [1, 3, 13] and a shift from traditional RPA to *Agentic Process Automation* (APA) [10, 14], where workflows are dynamically LLM-guided. However, relying on LLMs without explicit task semantics yields unreliable orchestration. Recent multi-agent systems (MAS) [8, 9, 12, 15] distribute roles across collaborating agents, yet orchestration remains implicit and tightly coupled to LLM-centric agents, offering limited support for heterogeneous components (reactive agents, statistical models, symbolic processes). While individual agent reasoning has improved [2, 4], the coordination layer for task decomposition and interaction remains a bottleneck for reliability and human oversight as MAS scale.

Why existing tools fall short: AgentFlow [11] and AutoGen [15] enable agent communication but lack semantic task understanding and heterogeneous coordination, and ADK [7] supports delegation while assuming homogeneous LLM agents that hinder integration of specialized non-LLM components.

Our approach: We present KiMO, a sub-component of DYNO [5] that makes orchestration explicit via *Knowledge Infusion* (KI) [6]. KiMO applies KI at two levels—(1) *planning ontologies* capturing expert hierarchical task decompositions and (2) *agent registries* specifying capabilities and I/O for heterogeneous components—enabling a two-stage pipeline where *PlanGen* produces ontology-grounded sub-task plans and *AgentGen* assembles registry-grounded workflows with capability-matched agents. This supports flexible orchestration of heterogeneous agents (e.g., LLM-based, statistical, symbolic) with human-inspectable coordination decisions (Figure 1).

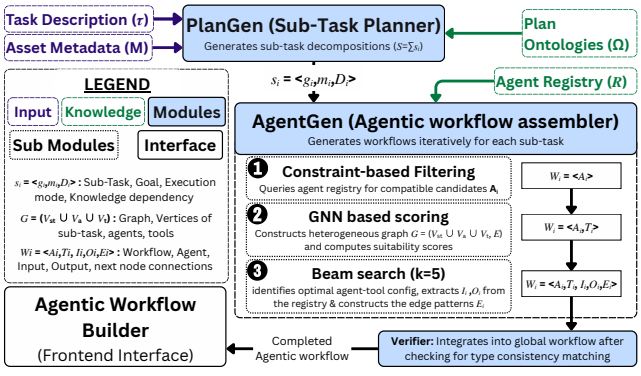

Figure 1: System Architecture

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/WTIF5096

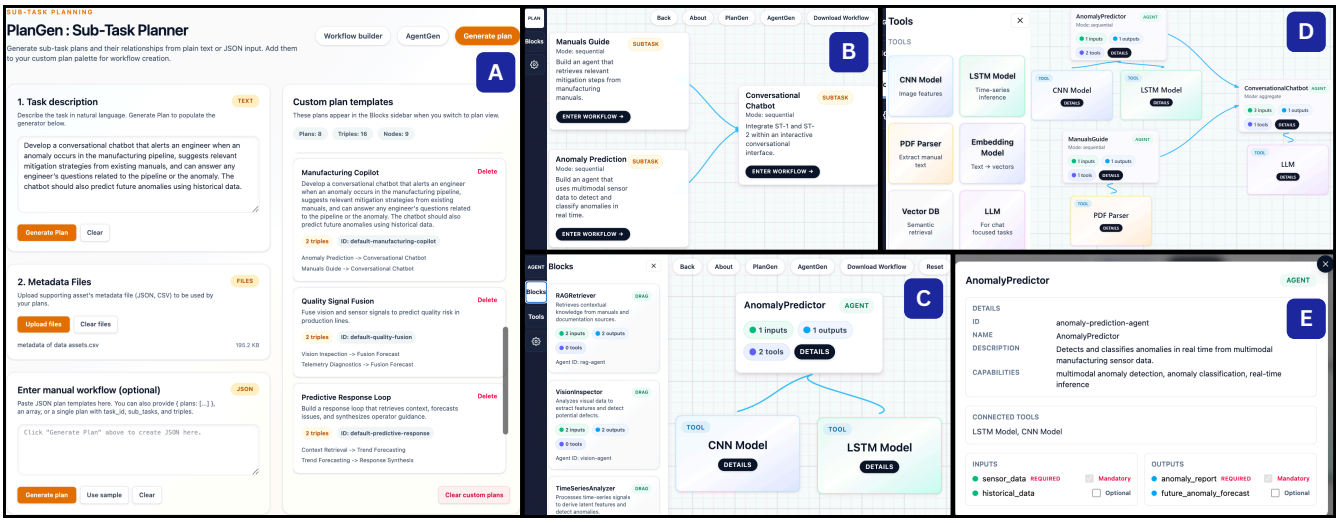

Figure 2: Real-time workflow construction for manufacturing anomaly detection: (A) PlanGen interface accepts task description and metadata; (B) Generated sub-task plan with dependencies; (C) AgentGen constructs agentic workflow for anomaly prediction sub-task, showing selected agent (AnomalyPredictor) and connected tools; (D) Complete manufacturing pipeline in workflow builder canvas with all agents and data streams; (E) Agent specification panel showing capabilities, I/O schemas, and tool connections. Note: Users can modify at any stage as changes propagate through Verifier before integration.

2 SYSTEM OVERVIEW

KiMO infuses structured knowledge through a two-stage pipeline: planning ontologies (Ω) guide task decomposition (Stage 1), while agent registry ($\mathcal{R}$) constrains workflow assembly (Stage 2) (Figure 1).

Stage 1: Ontology-guided Sub-task Planning (PlanGen). Given task description τ and metadata $\mathcal{M}$ (Figure 2A), PlanGen generates sub-task decompositions $P = s_1, \ldots, s_n$ where $s_i = \langle g_i, m_i, D_i \rangle$ specifies goal, execution mode (sequential/parallel), and knowledge dependencies. PlanGen employs a fine-tuned LLM (Llama 3.2 3B with LoRA) conditioned on plan ontologies (Ω) via retrieval augmented fine-tuning (RAFT) [16]. *Example:* For τ="detect manufacturing defects", ontology Ω containing hierarchical task structures yields s_1=⟨collect sensor data, sequential, $\emptyset$⟩, s_2=⟨preprocess signals, sequential, D_1⟩, s_3=⟨run CNN inference, parallel, D_2⟩, and decompositions P are rendered on an interactive canvas (Figure 2B) where experts can modify goals, dependencies, or execution modes.

Stage 2: Registry-guided Workflow Assembly (AgentGen). For each sub-task s_i received from PlanGen, AgentGen iteratively constructs workflow $w_i = \langle A_i, T_i, I_i, O_i, E_i \rangle$ specifying agents, tools, input/output data streams, and inter-agent connections through three registry-guided phases (Figure 1): **(i) Constraint-based Filtering** queries agent registry $\mathcal{R}$ for compatible candidates $A_{\text{cand}} = a \in \mathcal{R} \mid \text{capability}(a) \cap \text{required}(s_i) \neq \emptyset$ based on sub-task goals, yielding $w_i = \langle A_i \rangle$. **(ii) GNN-based Scoring** constructs heterogeneous graph $G = (V_{\text{subtask}} \cup V_{\text{agents}} \cup V_{\text{tools}}, E)$ over sub-tasks, agents, and tools, computing suitability scores via message passing over compatibility relations and historical profiles from $\mathcal{R}$, producing $w_i = \langle A_i, T_i \rangle$. *Example:* For s_3="run CNN inference", registry $\mathcal{R}$ contains AnomalyPredictor (type: cognitive, capability: CNN/LSTM, input: tensor, GPU: required) and StatisticalDetector (type: reactive, capability: threshold, input: scalar, GPU: false); GNN scores AnomalyPredictor=0.89 (capability match + GPU available) vs StatisticalDetector=0.34. **(iii) Beam Search** (width $k = 5$) identifies optimal agent-tool configurations and extracts I/O specifications I_i, O_i and constructs connection patterns E_i from registry metadata, completing $\langle A_i, T_i, I_i, O_i, E_i \rangle$ for a single sub-task (Figure 2C), with a Verifier validating type consistency before integration into the global workflow displayed in the Agentic Workflow Builder interface (Figure 2D) and executed workflows logged to retrain the GNN, enabling continuous improvement.

3 DEMONSTRATION PLATFORM

Our web-based platform enables interactive construction and execution of knowledge-infused multi-agent workflows through two integrated views (Figure 2).

Planning View (A,B): Users input task description τ and metadata $\mathcal{M}$ to invoke PlanGen's ontology-guided decomposition. Generated sub-tasks render as an interactive graph displaying execution modes and dependencies D_i. The interface supports real-time modification of decompositions and loading of custom templates, enabling domain experts to extend ontology Ω dynamically.

Agentic Workflow View (C,D,E): Selecting any sub-task s_i triggers AgentGen's three-phase assembly with live visualization of constraint filtering, GNN scoring, and beam search. Panel C displays the generated workflow graph with agent nodes, tool connections, and data streams for the Anomaly Prediction sub-task from the manufacturing pipeline. Panel E reveals detailed agent specifications from registry $\mathcal{R}$, including the AnomalyPredictor's capabilities, I/O schemas, and connected tools. Users can modify workflows or construct custom agents by selecting tools from registry $\mathcal{R}$ (Panel D left sidebar) and defining I/O specifications. The complete manufacturing pipeline workflow with all integrated agents and data streams is visualized in Panel D.

ACKNOWLEDGMENTS

This work is supported in part by NSF grant 2119654, "RII Track 2 FEC: Enabling Factory to Factory (F2F) Networking for Future Manufacturing".

REFERENCES

[1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. 2022. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691* (2022).

[2] Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, et al. 2025. Why do multi-agent llm systems fail? *arXiv preprint arXiv:2503.13657* (2025).

[3] Liying Cheng, Xingxuan Li, and Lidong Bing. 2023. Is gpt-4 a good data analyst? *arXiv preprint arXiv:2305.15038* (2023).

[4] Mingyan Gao, Yanzi Li, Banruo Liu, Yifan Yu, Phillip Wang, Ching-Yu Lin, and Fan Lai. 2025. Single-agent or Multi-agent Systems? Why Not Both? *arXiv preprint arXiv:2505.18286* (2025).

[5] Ritvik Garimella, Chathurangi Shyalika, Renjith Prasad Kaippilly Mana, and Amit Sheth. 2026. DYNO: Dynamic Neurosymbolic Orchestrator for Multi-Agent Systems. In *LLM-based Multi-Agent Systems: Towards Responsible, Reliable, and Scalable Agentic Systems*.

[6] Manas Gaur, Kalpa Gunaratna, Shreyansh Bhatt, and Amit Sheth. 2022. Knowledge-Infused Learning: A Sweet Spot in Neuro-Symbolic AI. *IEEE Internet Computing* 26, 4 (2022), 5–11. https://doi.org/10.1109/MIC.2022.3179759

[7] Google. 2025. Agent Development Kit (ADK). https://google.github.io/adk-docs/. Software framework for building and deploying AI agents.

[8] Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, et al. 2023. MetaGPT: Meta programming for a multi-agent collaborative framework. In *The Twelfth International Conference on Learning Representations*.

[9] Mengkang Hu, Yuhang Zhou, Wendong Fan, Yuzhou Nie, Bowei Xia, Tao Sun, Ziyu Ye, Zhaoxuan Jin, Yingru Li, Qiguang Chen, et al. 2025. Owl: Optimized workforce learning for general multi-agent assistance in real-world task automation. *arXiv preprint arXiv:2505.23885* (2025).

[10] Zelong Li, Shuyuan Xu, Kai Mei, Wenyue Hua, Balaji Rama, Om Raheja, Hao Wang, He Zhu, and Yongfeng Zhang. 2024. Autoflow: Automated workflow generation for large language model agents. *arXiv preprint arXiv:2407.12821* (2024).

[11] Zhuofeng Li, Haoxiang Zhang, Seungju Han, Sheng Liu, Jianwen Xie, Yu Zhang, Yejin Choi, James Zou, and Pan Lu. 2025. In-the-Flow Agentic System Optimization for Effective Planning and Tool Use. https://doi.org/10.48550/arXiv.2510.05592 arXiv:2510.05592 [cs.AI]

[12] OpenAI. 2025. API Agents | OpenAI. https://openai.com/agent-platform/. Accessed: 2025-11-02.

[13] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. 2023. Chatdev: Communicative agents for software development. *arXiv preprint arXiv:2307.07924* (2023).

[14] Michael Wornow, Avanika Narayan, Krista Opsahl-Ong, Quinn McIntyre, Nigam H Shah, and Christopher Re. 2024. Automating the enterprise with foundation models. *arXiv preprint arXiv:2405.03710* (2024).

[15] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. 2024. Autogen: Enabling next-gen LLM applications via multi-agent conversations. In *First Conference on Language Modeling*.

[16] Tianjun Zhang, Shishir G Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E Gonzalez. 2024. Raft: Adapting language model to domain specific rag. *arXiv preprint arXiv:2403.10131* (2024).