AGENTPbD: Interactive Agentic Workflow Generation from User Demonstration on Web Browsers

Jiawen Li[1], Zheng Ning[2], Yuan Tian[3], Toby Jia-Jun Li[2]
[1]University of Michigan, Ann Arbor, Michigan
[2]University of Notre Dame, Notre Dame, Inidana
[3]Purdue University, West Lafayette, Inidana
Email: lijiawen@umich.edu, zning@nd.edu, tian211@purdue.edu, toby.j.li@nd.edu

***Abstract*—Programming by Demonstration (PbD) enables users to automate tasks through examples, but traditional systems generate low-level scripts that are hard to generalize or reuse. Recent advances in Large Language Models (LLMs) offer the potential to infer higher-level task structures, but rely on ambiguous natural language input. We present AgentPbD, a system that synthesizes task-level agentic workflows from a single user demonstration. By capturing browser actions and contextual metadata, AgentPbD automatically infers user goals and intentions, transforming user demonstrations into an editable, modular LLM agent workflow, and displays it on the browser extension interface. Users can further review and modify the workflow through visual programming. We demonstrate how AgentPbD bridges PbD and LLM planning, enabling interpretable and generalizable automation of complex web tasks.**

***Index Terms*—End-user programming, Programming by demonstration, Web automation, LLM agents**

I. INTRODUCTION

User demonstrations provide explicit examples of how users perform tasks, and prior research in Programming by Demonstration (PbD) has shown its effectiveness in enabling non-programmers to build task automations [1]–[4]. However, traditional PbD systems often require multiple demonstrations from users for disambiguation, and are struggling in generalizing to similar tasks [5]–[7]. Furthermore, these systems often focus on generating the program based solely on the user's actions, without accounting for the preferences, knowledge, and behaviors encoded deeper in the demonstration flow, due to the difficulty of modeling such information with limited demonstration traces. However, this limitation hinders their performance on tasks that involve longer steps and more implicit user goals.

Recent advances in LLMs have demonstrated their strong capabilities in understanding tasks and user goals [8]. LLM-based agents are trained on large-scale dataset and enhanced by agentic workflows–modular sequences that integrate prompts, operations, and tools. These agents can handle complex tasks by reasoning about user intent, planning across multiple steps, and filling in missing details using commonsense knowledge [9]–[12]. However, current LLM agents rely heavily on natural language instructions, which are often vague or underspecified, especially for abstract, multi-step workflows that are hard to express precisely in language alone [13].

Recognizing the complementary strengths of LLMs and user demonstrations, we present AGENTPbD, an interactive system that automatically generates agentic workflows from a single user demonstration of their workflow on a web browser. Based on the procedural knowledge embedded in a task demonstration, AGENTPbD automatically parses the workflow into inter-connected sub-tasks as a tree structure, while allowing users to change tool use, modify prompts as well as reconstruct the workflow logic.

II. RELATED WORKS

Task automation has been widely explored in several PbD systems [3], [4], [7], [14]–[16]. These systems observe user demonstrations and generate scripts that replay and generalize actions across similar website elements. Rousillon [4] renders these scripts using a Scratch-style syntax, while MIWA [3] describes script behaviors in natural language. However, these systems rely on a static record-and-replay strategy, which limits their generalizability. TaskMind [7] advances this direction by capturing cognitive traces of user actions and organizing them into flow-based action sequences. Yet, its execution remains constrained to linear action flows. With the rise of LLMs, users can now use low-code platforms such as Dify [17], Coze [18], and n8n [19] to build customized workflow-based agents with visual, block-based interfaces. While these tools lower the entry barrier, they still assume users can manually translate their goals into logical steps [20]. More recently, automatic agent workflow generation and optimization methods [9], [21], [22] have been proposed to synthesize workflows from task requests. However, these systems are largely confined to closed-domain, static tasks (e.g., Q&A or math problems) and have yet to address open-ended, interactive web environments.

III. SYSTEM DESIGN

As shown in Fig. 1, AgentPbD transforms user demonstrations into editable workflows through three stages: collecting browser actions with context, generating task-level workflows via a multi-agent backend, and presenting the result in an interactive tree interface for easy customization. AgentPbD generates *task-level* workflows, so that they can be better generalized in future similar tasks, and users can perceive and edit high-level, abstract sub-tasks rather than tedious stepwise actions.

A. User Demonstration Collection

To infer users' implicit task goals and intentions, we collect a set of common browser *actions*, including clicks, text selection, text input, form submissions, and URL navigations. Alongside each action, we record the *contextual metadata*, such as the class, tag, and value attributes of the corresponding DOM elements, to capture the semantic implications of the interaction. To avoid user noises of meaningless, repeated actions, we selectively log meaningful user actions by filtering low-information elements (e.g., background containers, generic layout divs), postponing event capture to avoid transient input noise, and recording only finalized interactions with minimal contextual metadata.

We do **not** rely on visual cues like element screenshots. In our observations, textual context provides sufficient semantic information, while visual models often introduce latency and complexity (e.g., in systems like Operator). We also omit full-page content, as we find that action-context pairs are enough for concise and effective representation of user intent and attention. By combining user actions with their immediate structural context, we collect a compact and expressive demonstration record for generating high-level, generalizable workflows.

B. Task-level Workflow Generation

We designed a lightweight multi-agent system that takes user interaction logs as input and outputs a structured workflow in JSON format. The system consists of four agents: the **action** agent, the **context** agent, the **synthesizer** agent, and the **evaluator** agent. The action agent analyzes interaction data and infers tool usage based on a predefined set of tools commonly found in Model Context Protocal (MCP) and agent-based applications. In parallel, the context agent analyzes surrounding metadata (e.g., page content, element labels, UI states) to infer user goals, interests, and intentions. The results from both the action and context agents are then passed to the synthesizer agent, which aligns them to generate a goal-driven, abstracted workflow that captures both the intent and structure of the user's task. The evaluator agent subsequently scores the workflow based on alignment quality and abstraction level, considering factors such as the number of nodes and the specificity of each prompt.

As the user continues interacting with the system, the workflow dynamically evolves to reflect new actions and intentions. While the generated workflow may not be perfect at the beginning, it will progressively adapt to user intent and task context as users engage with it more. Furthermore, if the generated workflow does not meet the user's expectations, the user can directly refine by editing it via the interface.

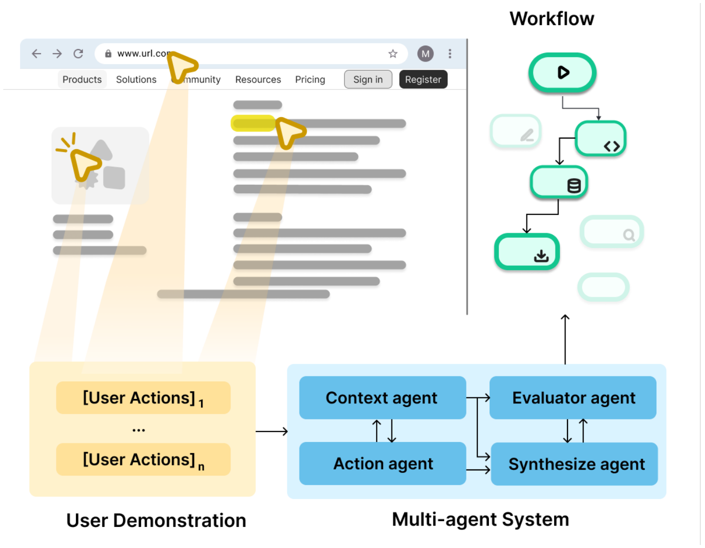


Fig. 1: System overview of AgentPbD. User demonstrations are collected as input for the multi-agent system, which dynamically generates task workflows that can be reused and generalized.

C. Interface

The UI for the generated workflow is shown in Fig. 2. When the user is operating in the browswer, the visualization of workflow will be displayed in the side panel. Each node represents a configurable LLM operation including the description, prompt, and tools. Edges between nodes indicate data flow or logical dependencies. By observing the workflow visualization, users can easily understand and grasp the functionality and intent of the workflow at the task level. Users have full control over the generated workflow: they can manually add new nodes on the existing graph, modify node configurations such as prompts, reconnect the nodes, or delete existing nodes or edges. Finally, as the user feels satisfied, they can encapsulate and save the entire workflow as a JSON file for future use.

IV. Presentation and Implications

We will showcase AgentPbD through an interactive live demonstration along with a poster during the conference. Attendees will have the opportunity to interact directly with AgentPbD, experiencing how workflow can be generated from demonstrations. This work explores user intentions embedded in actions, using task-level workflows as the core representation. As the next step, we are going to support the execution of workflows and conduct a user study to evaluate the system's usability and to understand user preferences and behaviors when using the system.

References

[1] B. A. Myers, “Visual programming, programming by example, and program visualization: a taxonomy,” *SIGCHI Bull.*, vol. 17, no. 4, p. 59–66, Apr. 1986. [Online]. Available: https://doi.org/10.1145/22339.22349

[2] B. Myers, “Demonstrational interfaces: A step beyond direct manipulation,” *Computer*, vol. 25, no. 8, pp. 61–73, 1992.

[3] W. Chen, X. Liu, J. Zhang, I. I. Lam, Z. Huang, R. Dong, X. Wang, and T. Zhang, “Miwa: Mixed-initiative web automation for better user control and confidence,” in *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST ’23. New York, NY, USA: Association for Computing Machinery, 2023. [Online]. Available: https://doi.org/10.1145/3586183.3606720

[4] S. E. Chasins, M. Mueller, and R. Bodik, “Rousillon: Scraping distributed hierarchical web data,” in *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, ser. UIST ’18. New York, NY, USA: Association for Computing Machinery, 2018, p. 963–975. [Online]. Available: https://doi.org/10.1145/3242587.3242661

[5] A. L. Ambler and Y.-T. Hsia, “Generalizing selection in by-demonstration programming,” *Journal of Visual Languages Computing*, vol. 4, no. 3, pp. 283–300, 1993. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1045926X83710177

[6] G. W. Paynter, “Generalizing programming by demonstration,” in *Proceedings of the 6th Australian Conference on Computer-Human Interaction (OZCHI ’96)*, ser. OZCHI ’96. USA: IEEE Computer Society, 1996, p. 344.

[7] Y. Yin, Y. Mei, C. Yu, T. J.-J. Li, A. K. Jadoon, S. Cheng, W. Shi, M. Chen, and Y. Shi, “From operation to cognition: Automatic modeling cognitive dependencies from user demonstrations for gui task automation,” in *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, ser. CHI ’25. New York, NY, USA: Association for Computing Machinery, 2025. [Online]. Available: https://doi.org/10.1145/3706598.3713356

[8] R. Y. Pang, K. J. K. Feng, S. Feng, C. Li, W. Shi, Y. Tsvetkov, J. Heer, and K. Reinecke, “Interactive Reasoning: Visualizing and Controlling Chain-of-Thought Reasoning in Large Language Models,” Jun. 2025, arXiv:2506.23678 [cs]. [Online]. Available: http://arxiv.org/abs/2506.23678

[9] X. Tan, B. Li, X. Qiu, C. Qu, W. Chu, Y. Xu, and Y. Qi, “Meta-agent workflow:streamlining tool usage in llms through workflow construction, retrieval, and refinement,” in *Companion Proceedings of the ACM on Web Conference 2025*. Sydney NSW Australia: ACM, May 2025, pp. 458–467. [Online]. Available: https://dl.acm.org/doi/10.1145/3701716.3715247

[10] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, W. X. Zhao, Z. Wei, and J. Wen, “A survey on large language model based autonomous agents,” *Frontiers of Computer Science*, vol. 18, no. 6, Mar. 2024. [Online]. Available: http://dx.doi.org/10.1007/s11704-024-40231-1

[11] J. Zhang, J. Xiang, Z. Yu, F. Teng, X. Chen, J. Chen, M. Zhuge, X. Cheng, S. Hong, J. Wang, B. Zheng, B. Liu, Y. Luo, and C. Wu, “AFlow: Automating Agentic Workflow Generation,” Apr. 2025, arXiv:2410.10762 [cs]. [Online]. Available: http://arxiv.org/abs/2410.10762

[12] J. Zhang and I. Arawjo, “ChainBuddy: An AI-assisted Agent System for Generating LLM Pipelines,” in *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. Yokohama Japan: ACM, Apr. 2025, pp. 1–21. [Online]. Available: https://dl.acm.org/doi/10.1145/3706598.3714085

[13] M. Pickering, “How Humans Communicate Programming Tasks in Natural Language and Implications For End-User Programming with LLMs,” 2025.

[14] R. Krosnick and S. Oney, “Understanding the challenges and needs of programmers writing web automation scripts,” in *2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2021, pp. 1–9.

[15] K. Pu, R. Fu, R. Dong, X. Wang, Y. Chen, and T. Grossman, “Semanticon: Specifying content-based semantic conditions for web automation programs,” in *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST ’22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3526113.3545691

[16] S. Chasins, S. Barman, R. Bodik, and S. Gulwani, “Browser record and replay as a building block for end-user web automation tools,” in *Proceedings of the 24th International Conference on World Wide Web*, ser. WWW ’15 Companion. New York, NY, USA: Association for Computing Machinery, 2015, p. 179–182. [Online]. Available: https://doi.org/10.1145/2740908.2742849

[17] A. Leatherwood and V. Matta, “Workshop: Building ai applications with dify.ai: A hands-on workshop,” in *MWAIS 2025 Proceedings*, 2025, p. 42, workshop introduction to the open-source Dify.ai low-code platform.

[18] “Coze,” https://www.coze.com/, [Accessed 20-07-2025].

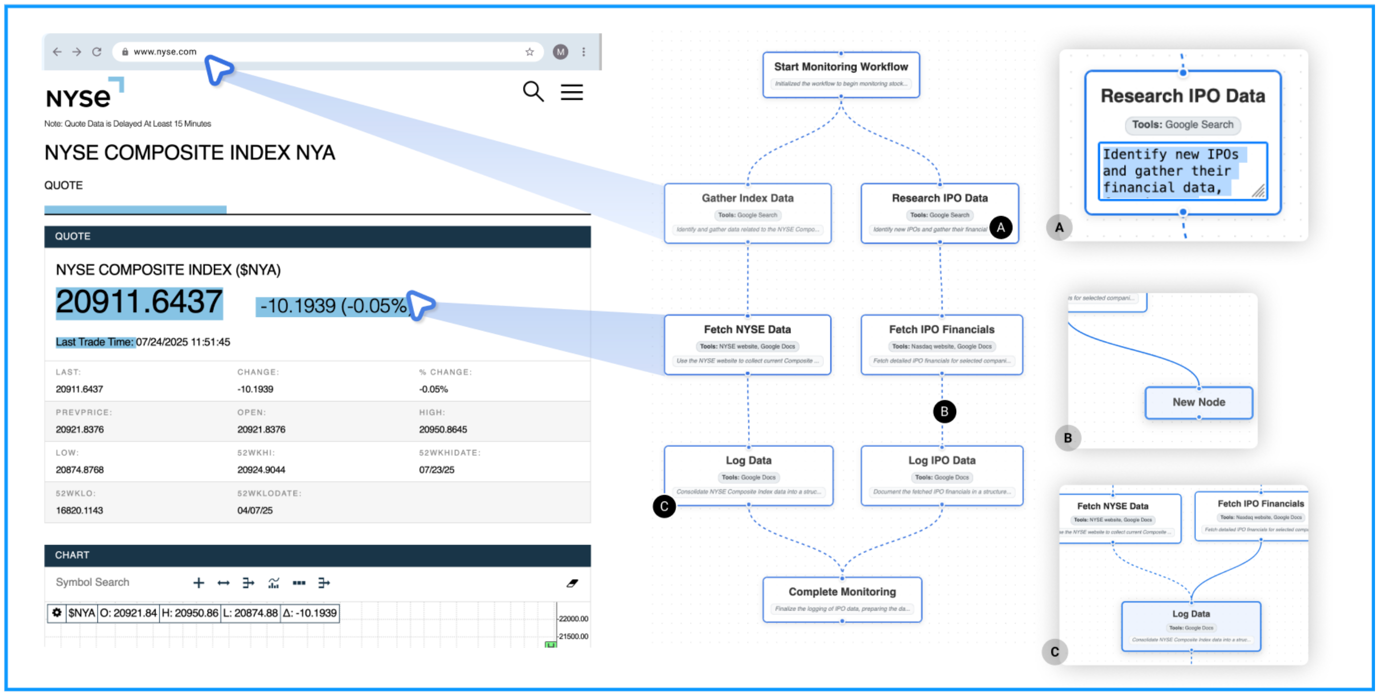
[19] “n8n,” https://n8n.io/, [Accessed 20-07-2025].

[20] A. J. Ko, B. A. Myers, and H. H. Aung, “Six learning barriers in end-user programming systems,” in *2004 IEEE Symposium on Visual Languages - Human Centric Computing*, 2004, pp. 199–206.

[21] X. Zhang, R. Dou, E. Hu, M. Zhao, Y. Ding, and Z. Dou, “Collaborative Optimization Approach for Workflow Agents in User Behavior Modeling,” in *Companion Proceedings of the ACM on Web Conference 2025*. Sydney NSW Australia: ACM, May 2025, pp. 2988–2992. [Online]. Available: https://dl.acm.org/doi/10.1145/3701716.3719228

[22] Z. Li, S. Xu, K. Mei, W. Hua, B. Rama, O. Raheja, H. Wang, H. Zhu, and Y. Zhang, “Autoflow: Automated workflow generation for large language model agents,” 2024. [Online]. Available: https://arxiv.org/abs/2407.12821

Appendix


www.nyse.com
NYSE
Note: Quote Data is Delayed At Least 15 Minutes
NYSE COMPOSITE INDEX NYA
QUOTE
QUOTE
NYSE COMPOSITE INDEX ($NYA)
20911.6437
-10.1939 (-0.05%)
Last Trade Time: 07/24/2025 11:51:45
CHART
Symbol Search
$NYA O: 20921.84 H: 20950.86 L: 20874.88 Δ: -10.1939
22000.00
21500.00
Start Monitoring Workflow
Initialized the workflow to begin monitoring stock...
Gather Index Data
Tools: Google Search
Identify and gather data related to the NYSE Compo...
Research IPO Data
Tools: Google Search
Identify new IPOs and gather their financial
Fetch NYSE Data
Tools: NYSE website, Google Docs
Use the NYSE website to collect current Composite ...
Fetch IPO Financials
Tools: Nasdaq website, Google Docs
Fetch detailed IPO financials for selected compani...
Log Data
Tools: Google Docs
Consolidate NYSE Composite Index data into a struc...
Log IPO Data
Tools: Google Docs
Document the fetched IPO financials in a structure...
Complete Monitoring
Finalize the logging of IPO data, preparing the da...
A
B
C
Research IPO Data
Tools: Google Search
Identify new IPOs and gather their financial data,
New Node
Fetch NYSE Data
Fetch IPO Financials
Log Data

LAST:	CHANGE:	% CHANGE:
20911.6437	-10.1939	-0.05%
PREVPRICE:	OPEN:	HIGH:
20921.8376	20921.8376	20950.8645
LOW:	52WKHI:	52WKHIDATE:
20874.8768	20924.9044	07/23/25
52WKLO:	52WKLODATE:	
16820.1143	04/07/25	

Fig. 2: The UI of AgentPbD. **(A)** Users can edit the prompt of the node agent. **(B)** Users can add and define a new node agent into the workflow. **(C)** Users can delete and reconstruct the workflow logic.