Participant Survey

This study explores how well users can understand the behavior of LLM agents for software development , using a new debugging tool compared to traditional log files.

1. Participant Key

_____

2. Age Range

Mark only one oval.

- ◯ 18-24
- ◯ 25-34
- ◯ 35-44
- ◯ 45+

3. Education Level

Mark only one oval.

- ◯ High school
- ◯ Bachelor's
- ◯ Master's
- ◯ PhD

Technical Experience

4. Years of Programming Experience

Mark only one oval.

- ◯ None
- ◯ 1-3 years
- ◯ 3-5 years
- ◯ 5+ years

5. Familiarity with LLM Agents

We don't mean ChatGPT! *We mean tools that use large language models (LLM) to perform complex tasks, reason, and interact with its environment far beyond the scope of a simple query. They are often used for workflow automation.*

Mark only one oval.

- ◯ None
- ◯ Basic (heard of them)
- ◯ Moderate (used them)
- ◯ Advanced (developed them or analyzed them)

6. Experience with LLM Agents and Frameworks

If you have experience in this area, please list all agents and frameworks you are familiar with.

_____

7. What text editor are you the most familiar with?

This is the editor we will try to make available for you when completing the tasks.

Mark only one oval.

- ◯ VS Code
- ◯ Notepad++
- ◯ Vim
- ◯ Emacs
- ◯ Other: _____

8. Would you like to receive feedback on the tasks?

Mark only one oval.

- Yes
- No

Trajectory Comprehension Task

This study explores how well users can understand the behavior of LLM agents for software development, using a new debugging tool compared to traditional log files.

Your Task

You will analyze a log of an LLM agent used for some aspect of software development. The log details the agent's thoughts, actions, and results. You will answer a series of questions—some high-level (e.g., about the agent's goal) and some detailed (e.g., about specific actions or errors)—to test your understanding of the agent's behavior. You'll be assigned to use either our optimized debugger view or a raw text log to answer these questions.

The task should take approximately 20–30 minutes, and your answers will help us enhance tools for software developers.

1. Participant Key

_____

High-Level Questions

2. What is the primary goal of the agent?

For example, if analyzing a Pizza Preparing Agent, you might say: "The agent's goal is to prepare a pepperoni pizza by selecting the correct ingredients and baking it according to the recipe."

_____

_____

_____

_____

_____

3. Did the agent complete its task?

Mark only one oval.

- ◯ Yes, the task was completed successfully.
- ◯ No, the task was not completed.

4. What are the main action categories performed by the agent?

   For a Pizza Preparing Agent, you might list actions like “Select ingredients,” “Mix dough,” and “Bake pizza,” focusing on action types, not specific details.

5. What is the most frequently performed action?

6. What is the key challenge or issue the agent encounters during the task?

   *For a Pizza Preparing Agent, you might say “The agent has trouble correctly setting the temperature of the oven. The oven keeps returning a **OvenNotAvailable** error, indicating that it is not set up correctly." **Make sure to name the specific reoccurring error.***

7. Does the agent exhibit any repetitive behavior? If so, describe one example.

   *Describe an instance if there is one, where the repetitive behavior persists over at least **three** query-tool call cycles. E.g. "The agent moves the dough from the workspace to the oven, and back to the workspace 3 times in a row before turning on the oven".*

Detailed Questions

8. What specific changes does the agent make to the **cli.py** file when it **first** modifies it?

   Copy the diff of the change, or the tool call that makes the change.

9. Which specific files does the agent modify to implement the **income report generation** feature, and what changes are made to each?

   List all files that are modified to support this feature. Also copy and paste the specific lines that are modified.

10. What feedback does the agent receive after its **first** attempt to run the test script for the finance tracker, and how does it influence the next action?

    Explain, or copy and paste the results of the test script execution. Explain what the agent does next.

11. Which Python packages does the agent install over the course of the run?

    List each one.

12. What feedback does the agent receive from running the test script for the finance tracker **after it has solved all import issues**?

    Explain, or copy and paste the results of the test script execution.

13. Which file is the agent primarily focused on modifying **towards the end** of the run?

    Provide the name of the file.

14. Does the agent ever attempt to make the same change **twice** before reevaluating the tests? If so, describe one instance.

Copy the tool calls from this instance, if there is any.

15. Analyze the final four changes the agent makes to the repository. How do these changes relate to each other?

Describe any similarities or differences in these changes. Do they all serve the same purpose, or are they trying to accomplish different things?

Questions Regarding the Task Difficulty

Based on the NASA Task Load Index questionare.

16. Mental Demand

How mentally demanding was the task?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

17. Temporal Demand

How hurried or rushed was the pace of the task?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

18. Performance

How successful were you in accomplishing what you were asked to do?

Mark only one oval.

	1	2	3	4	5	6	7	
Perfe	◯	◯	◯	◯	◯	◯	◯	Failure

19. Effort

How hard did you have to work to accomplish your level of performance?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

20. Frustration

How insecure, discouraged, irritated, stressed, and annoyed were you?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

Agent Bug Localization Task

This study explores how well users can understand the behavior of LLM agents for software development, using a new debugging tool compared to traditional log files.

Your Task You will analyze the behavior of an LLM agent designed for software development to identify the likely causes of its failure to complete a given task. The goal is to pinpoint bugs in the agent's core code that led to the failure, rather than issues with the LLM's responses. Potential bugs may include misinterpretations of the LLM's outputs, incorrect logic in the agent's decision-making, or other implementation errors.

The task should take approximately 10–15 minutes, and your answers will help us enhance tools for software developers.

1. Participant Key

_____

Task Scenario

The Agent
RepairAgent is an autonomous, LLM-based agent designed for automated program repair. It leverages a large language model (LLM) to autonomously plan and execute actions to fix software bugs by invoking a set of specialized tools. Unlike traditional LLM-based repair techniques that rely on fixed prompts or hard-coded feedback loops, RepairAgent dynamically interacts with the codebase, mimicking a human developer's approach. It gathers information about the bug, searches for repair ingredients, proposes fixes, and validates them through test execution. The agent operates through a middleware that orchestrates communication between the LLM and tools, guided by a finite state machine with states e.g. *"Understand the bug," "Collect information to fix the bug," "Try to fix the bug,"* and *"Done."*

The Agent's Task
In the provided trajectory, RepairAgent attempts to fix issue CSV-75 in the Apache Commons CSV project involving inconsistent handling of end-of-line (EOL) characters.

Expected Agent Behavior
We expect RepairAgent to correctly identify the cause of the CSV-75 bug to be in the ExtendedBufferedReader class, where the read() method inconsistently increments the line counter only for \n (LF), while readLine() handles \r (CR), \n (LF), and \r\n (CRLF) as line terminators. The agent should autonomously use its tools (e.g., read_range, extract_tests, search_code_base) to analyze the codebase, recognize the discrepancy in EOL handling, and formulate a hypothesis. RepairAgent should then invoke the write_fix tool to generate a patch that modifies the read() method to check for \r or \n before incrementing lineCounter, and validate the fix by running the project's test suite. If the tests fail, the agent should reconsider its hypothesis and try other potential fixes.

2. Find and describe all potential bugs!

   Analyze the provided trajectory to determine why RepairAgent failed to fix the CSV-75 bug in the Apache Commons CSV project. Focus on potential issues in the agent's logic, such as misinterpretations of the LLM's outputs, incorrect logic in the agent's decision-making, or other implementation errors.

Questions Regarding the Task Difficulty

Based on the NASA Task Load Index questionare.

3. Mental Demand

How mentally demanding was the task?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

4. Temporal Demand

How hurried or rushed was the pace of the task?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

5. Performance

How successful were you in accomplishing what you were asked to do?

Mark only one oval.

	1	2	3	4	5	6	7	
Perfe	◯	◯	◯	◯	◯	◯	◯	Failure

6. Effort

How hard did you have to work to accomplish your level of performance?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

7. Frustration

How insecure, discouraged, irritated, stressed, and annoyed were you?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

Agent Bug Localization Task

This study explores how well users can understand the behavior of LLM agents for software development, using a new debugging tool compared to traditional log files.

Your Task You will analyze the behavior of an LLM agent designed for software development to identify the likely causes of its failure to complete a given task. The goal is to pinpoint bugs in the agent's core code that led to the failure, rather than issues with the LLM's responses. Potential bugs may include misinterpretations of the LLM's outputs, incorrect logic in the agent's decision-making, or other implementation errors.

The task should take approximately 10–15 minutes, and your answers will help us enhance tools for software developers.

1. Participant Key

_____

Task Scenario

The Agent
ExecutionAgent is an autonomous, LLM-based agent designed to set up and execute test suites of arbitrary software projects. It leverages a large language model (LLM) to autonomously plan and execute commands, mimicking a human developer's approach to configuring and testing projects. Unlike traditional automated testing pipelines that rely on hard-coded scripts or language-specific heuristics, ExecutionAgent dynamically interacts with the project's environment, using tools like terminal commands, file I/O, and web searches. It employs meta-prompting to generate up-to-date, technology-specific guidelines and iteratively refines its process based on command outputs. The agent operates through a control center that manages tool invocations and feedback loops, guided by a two-phase process: preparation (gathering project-specific information) and a feedback loop (iterating to install and test), producing scripts for reproducible test execution.

The Agent's Task
In the provided trajectory, ExecutionAgent attempts to set up and execute the test suite of the Gson project, a Java-based library for JSON serialization and deserialization, which uses Maven as its build system.

Expected Agent Behavior
We expect ExecutionAgent to correctly set up and execute the test suite of the Gson project, a Java-based library using Maven. The agent should autonomously use its tools (e.g., terminal, read_file, write_file) to clone the repository, analyze project files like pom.xml and README.md, and identify the mvn test command for running tests in src/test/java. ExecutionAgent should then produce a Dockerfile and an initialization script to create a sandbox environment with the appropriate dependencies for the project, execute the mvn test command within this environment, and monitor outputs to confirm successful test execution. If errors occur, the agent should iteratively refine its commands based on summarized outputs, ensuring accurate test result reporting and the generation of reproducible scripts.

2. Find and describe all potential bugs!

   Analyze the provided trajectory to determine why ExecutionAgent failed to set up and execute the test suite of the Gson project. Focus on potential issues in the agent's logic, such as misinterpretations of the LLM's outputs, incorrect logic in the agent's decision-making, or other implementation errors.

Questions Regarding the Task Difficulty

Based on the NASA Task Load Index questionnare.

3. Mental Demand

How mentally demanding was the task?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

4. Temporal Demand

How hurried or rushed was the pace of the task?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

5. Performance

How successful were you in accomplishing what you were asked to do?

Mark only one oval.

	1	2	3	4	5	6	7	
Perfe	◯	◯	◯	◯	◯	◯	◯	Failure

6. Effort

How hard did you have to work to accomplish your level of performance?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

7. Frustration

How insecure, discouraged, irritated, stressed, and annoyed were you?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very High

Agent Debugger Feedback Form

The task should take approximately 5 minutes, and your answers will help us enhance our tool for software developers.

1. How would you rate your overall experience with the LLM Agent Debugger tool?

   Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Excellent

2. What do you find most useful about the tool?

   _____

3. Are there any aspects of the tool that you find confusing or difficult to use?

   _____

   _____

   _____

   _____

   _____

User Interface

4. How would you rate the clarity and readability of the UI

   Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Excellent

5. Is the layout of the Agent Runs, Agent Activity, and Repository sections intuitive?
If not, please elaborate why not, and share any suggestions or remarks you might have.

6. Are the colors (e.g, blue for agent messages, green for responses) helpful in distinguishing actions and outcomes?
If not, please elaborate why not, and share any suggestions or remarks you might have.

7. Is the font size of the messages and other components comfortable to read?
If not, please elaborate why not, and share any suggestions or remarks you might have.

Functionality

8. How effective do you find the tool in visualizing agent actions and events?

Mark only one oval.

	1	2	3	4	5	6	7	
Not	◯	◯	◯	◯	◯	◯	◯	Very Effective

9. How would you rate the quality of the generated summaries in the message bubbles?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very Good

10. Are the summaries concise and helpful, or do they omit important details?
If not, please elaborate why not, and share any suggestions or remarks you might have.

Message Inspector Popup Window

11. How useful do you find the message inspector popup window for reviewing details?

Mark only one oval.

	1	2	3	4	5	6	7	
Not	◯	◯	◯	◯	◯	◯	◯	Very Useful

12. Are the window controls (toggle fullscreen, resize, drag, close) intuitive and easy to use?
If not, please elaborate why not, and share any suggestions or remarks you might have.

13. Do you find the ability to compare two messages in the message inspector useful?

If not, please elaborate why not, and share any suggestions or remarks you might have.

14. How clear and well-organized do you find the layout of JSON objects displayed in the message inspector?

Mark only one oval.

	1	2	3	4	5	6	7	
Very	◯	◯	◯	◯	◯	◯	◯	Very Good

15. Are there any improvements you would suggest for the message inspector popup window?

Suggested Improvements

16. Do you have any suggestions for improving the UI or making it more user-friendly not mentioned before?

17. Are there any bugs or issues you encountered while using the tool not mentioned before?

18. Please provide any other feedback or suggestions you have about the tool.