Machine Programming

Lecture 11 – Agentic Frameworks for Software Development (2)

Ziyang Li

Logistics – Week 6

- Assignment 2
  - https://github.com/machine-programming/assignment-2
  - Due this Sunday (Oct 5th)
  - Expected to take quite some time, so please start working on it early
- Oral presentation sign up sheet
  - Sending out today
  - Oral presentation starting on Week 8

The Course So Far

Behavioral Specification

- What should the program do?

1. Examples
2. Types
3. Functional Specifications
4. Natural Language

Synthesis Strategy

- How do we find such a program?

Enumeration

Language Models

- Prompting
- Iterative refinement
- Agentic frameworks

Structural Specification

- What is the space of the programs?

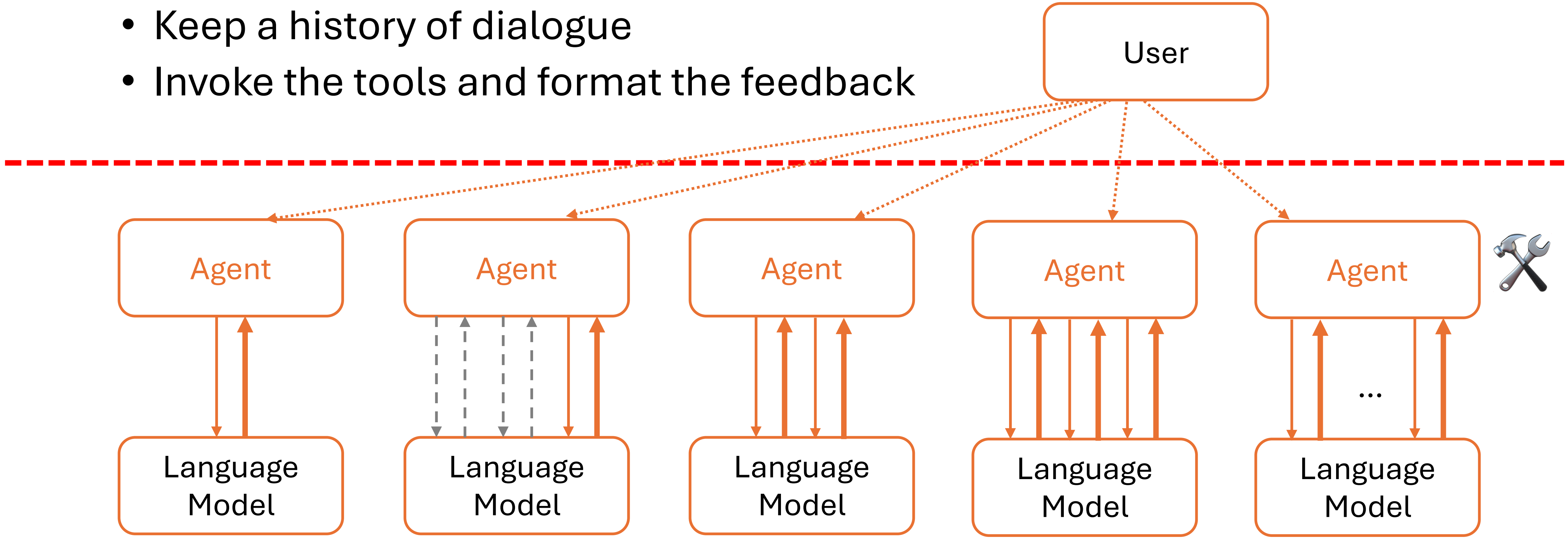
General Purpose Programming Language

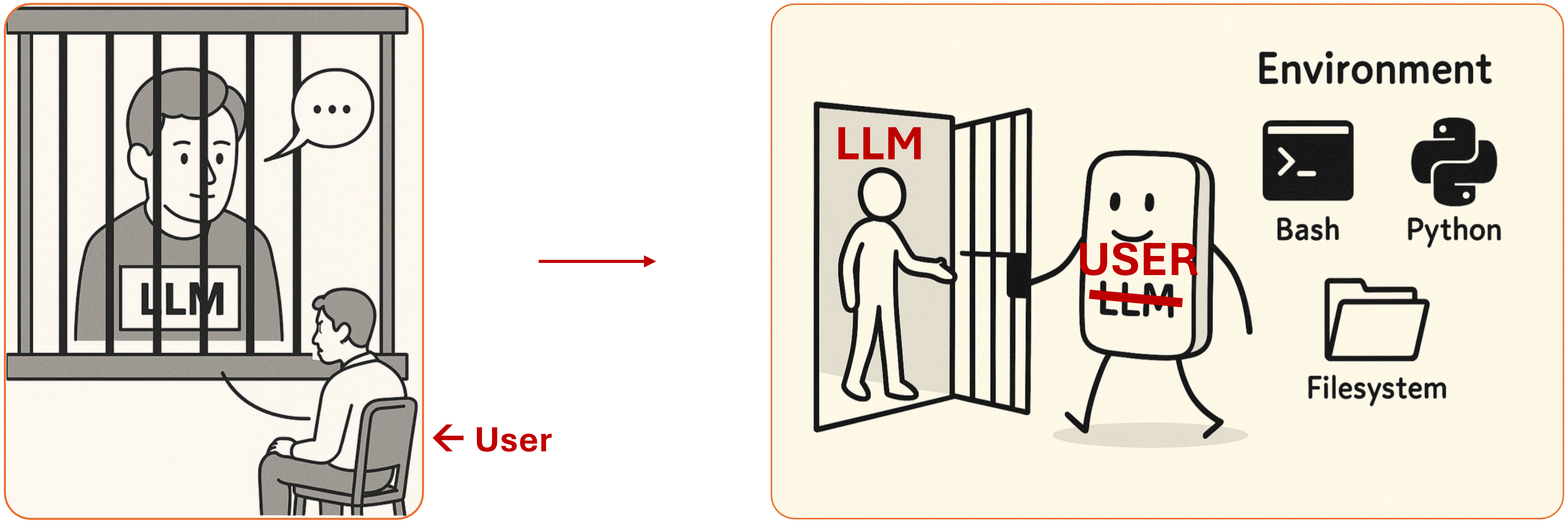
Python / Java / C / Rust / …

Domain Specific Languages

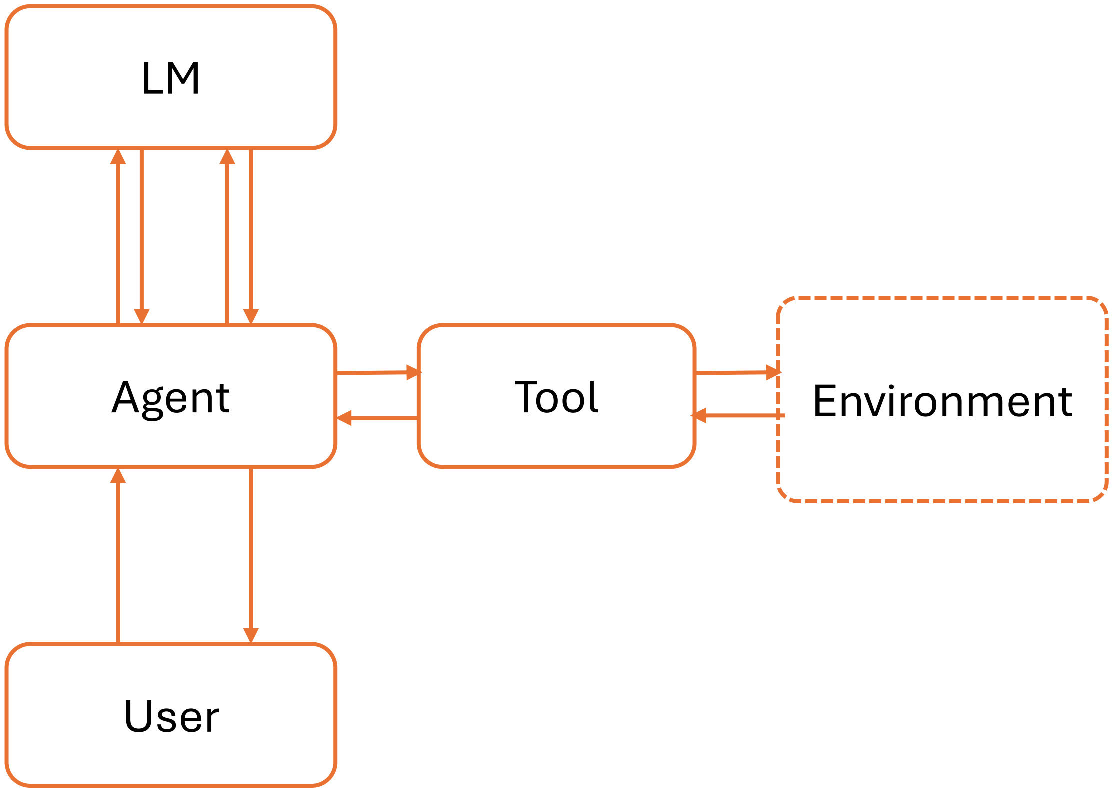
Agentic Pipelines

- Agent holds the responsibility to
  - Send requests to LLM
  - Keep a history of dialogue
  - Invoke the tools and format the feedback



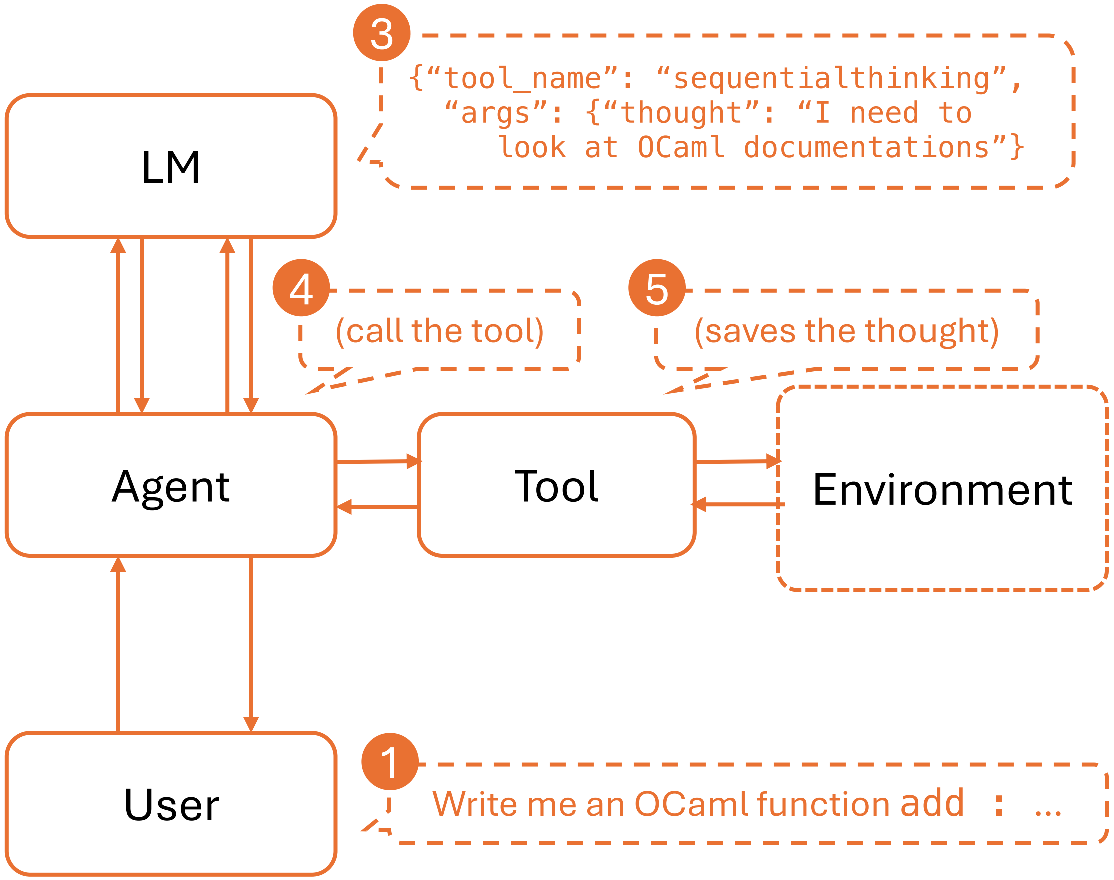

LLM
User
LLM
USER
~~LLM~~
Environment
Bash
Python
Filesystem

Agentic Framework



Sequential Thinking Tool

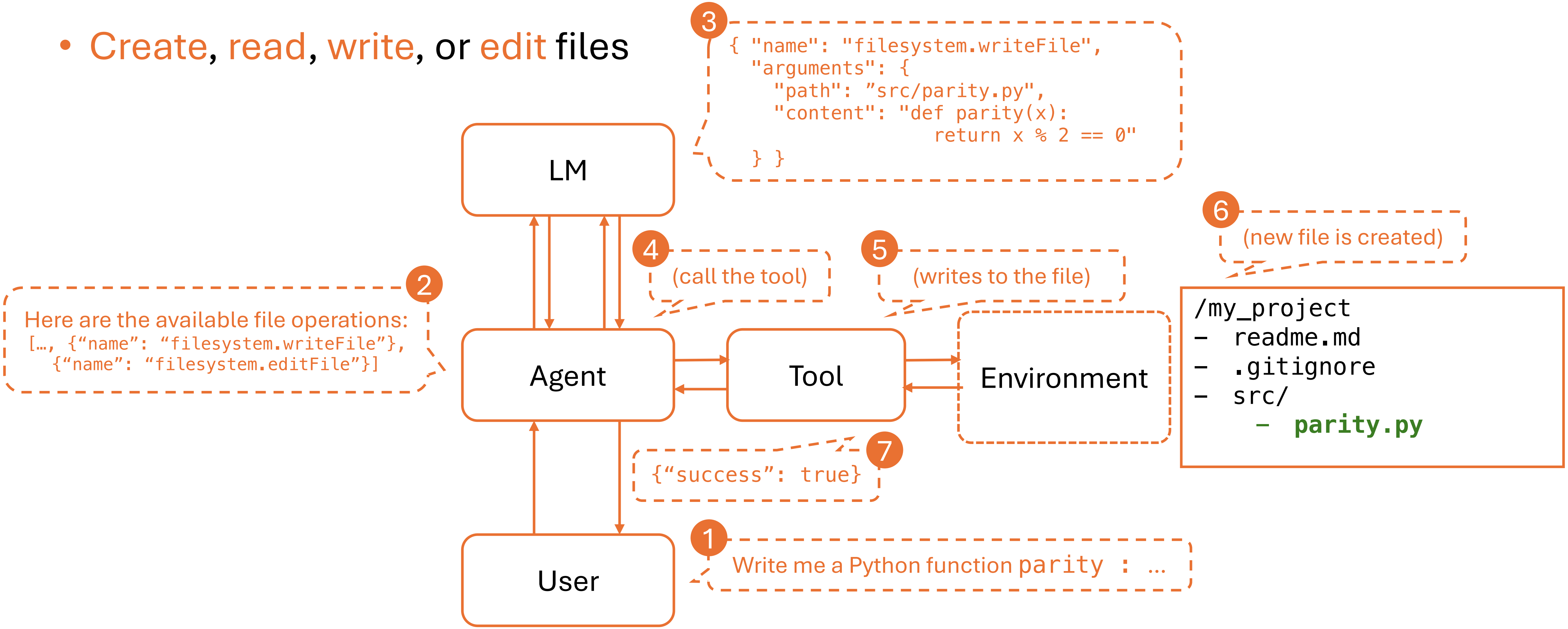
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you inten
- `thoughtNumber` (in
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need



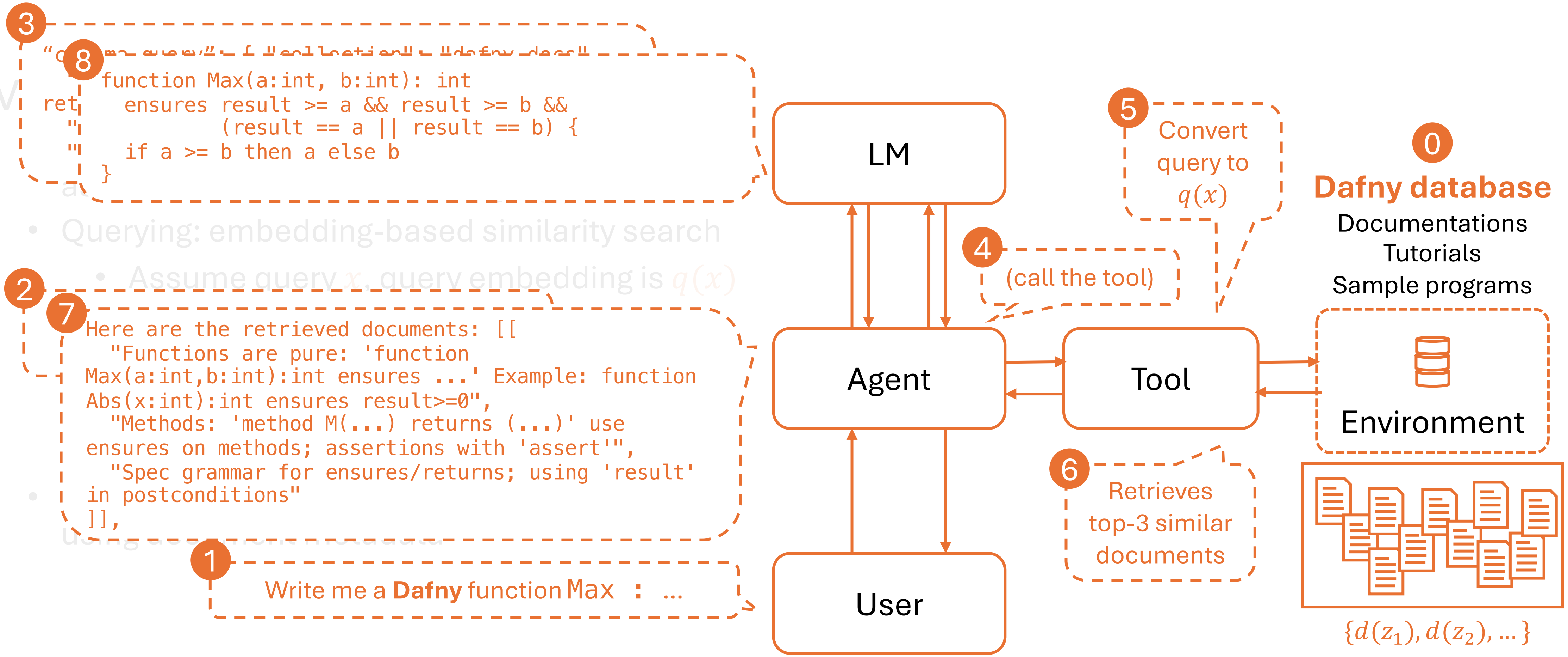
https://github.com/modelcontextprotocol/servers/blob/main/src/sequentialthinking

File System (FS) Tool

- Create, read, write, or edit files



RAG Database as Tool



Topics of Today

- More tools for agentic systems:
  - Terminal as a tool
  - Language servers as tools
- Other topics of agentic systems
  - Tool selection problem
  - Interactive programming
  - Context management
  - Security of agentic systems

Terminal as Tool

- Usage
  - Running compilers: `gcc foo.c` → syntax/type errors
  - Running programs: `python foo.c` → runtime errors
  - Running tests: `pytest` → test cases pass/fail
  - Managing packages: `pip` (Python), `cargo` (Rust), `npm` (JavaScript)
  - Processing files: `cat` (read), `grep`/`find` (search), `sed` (edit), `echo` (file write)
  - Managing folders and directories: `cd` (go to dir), `ls` (list items in dir)
- Interaction
  - Input: command (string)
  - Output: stdout / stderr (string), exit code (integer)

Now I need to update the test files to use the correct types. Let me update the test that creates generic types:

test_type_inference.rs +1 -1 •

Now let me try to compile again:

Auto-Ran command: cd, cargo check

```
$ cd /Users/liby99/Local/Projects/scallop-lang/scallop-v3 && cargo check --package scallop-fir
For more information about an error, try `rustc --explain E0277`.
warning: `scallop-fir` (lib) generated 1 warning
error: could not compile `scallop-fir` (lib) due to 4 previous errors; 1 warning emitted
```

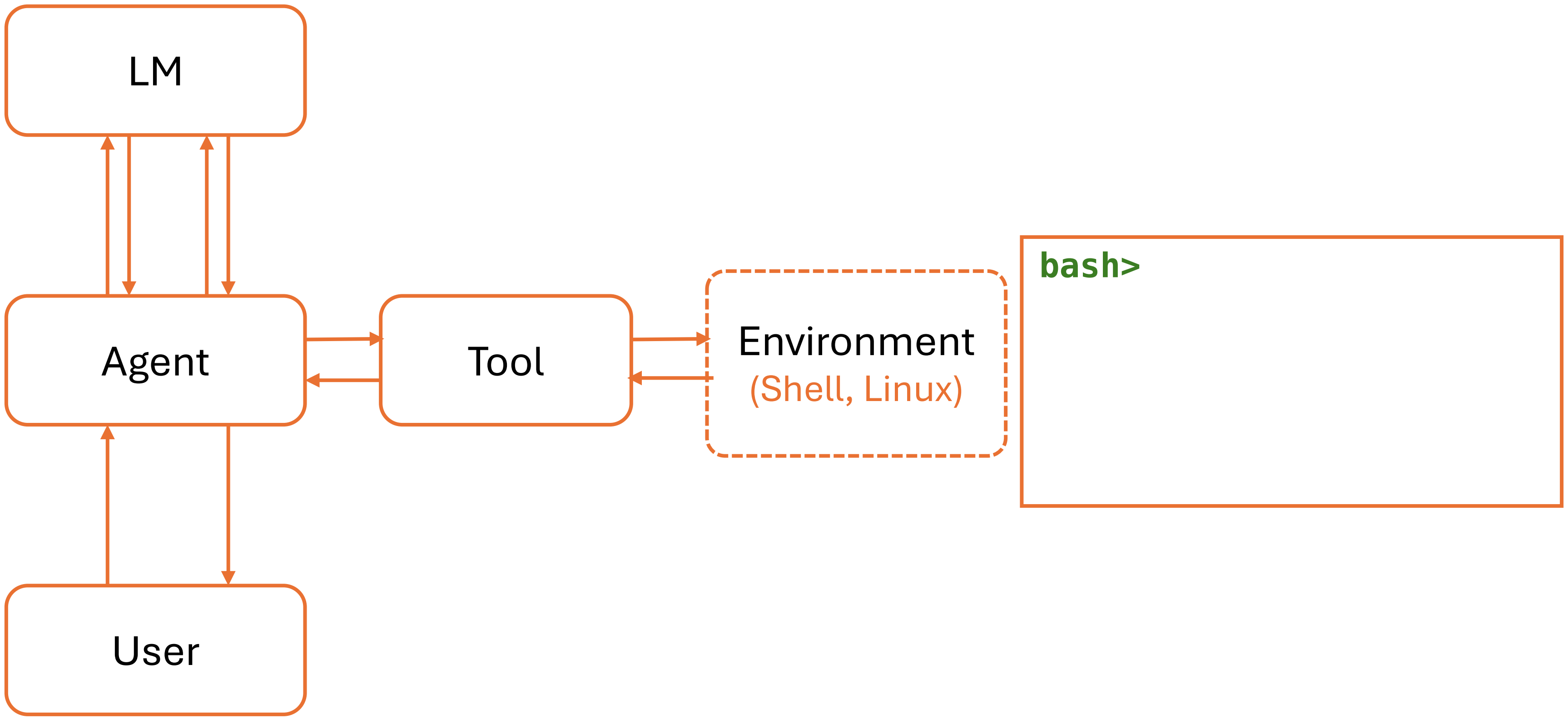
I see the issue. The problem is that I'm trying to match on `Type::Named` and `Type::Generic` but the AST `Type` is actually a struct, not an enum. Let me check the actual structure of the AST Type:

Read types.rs

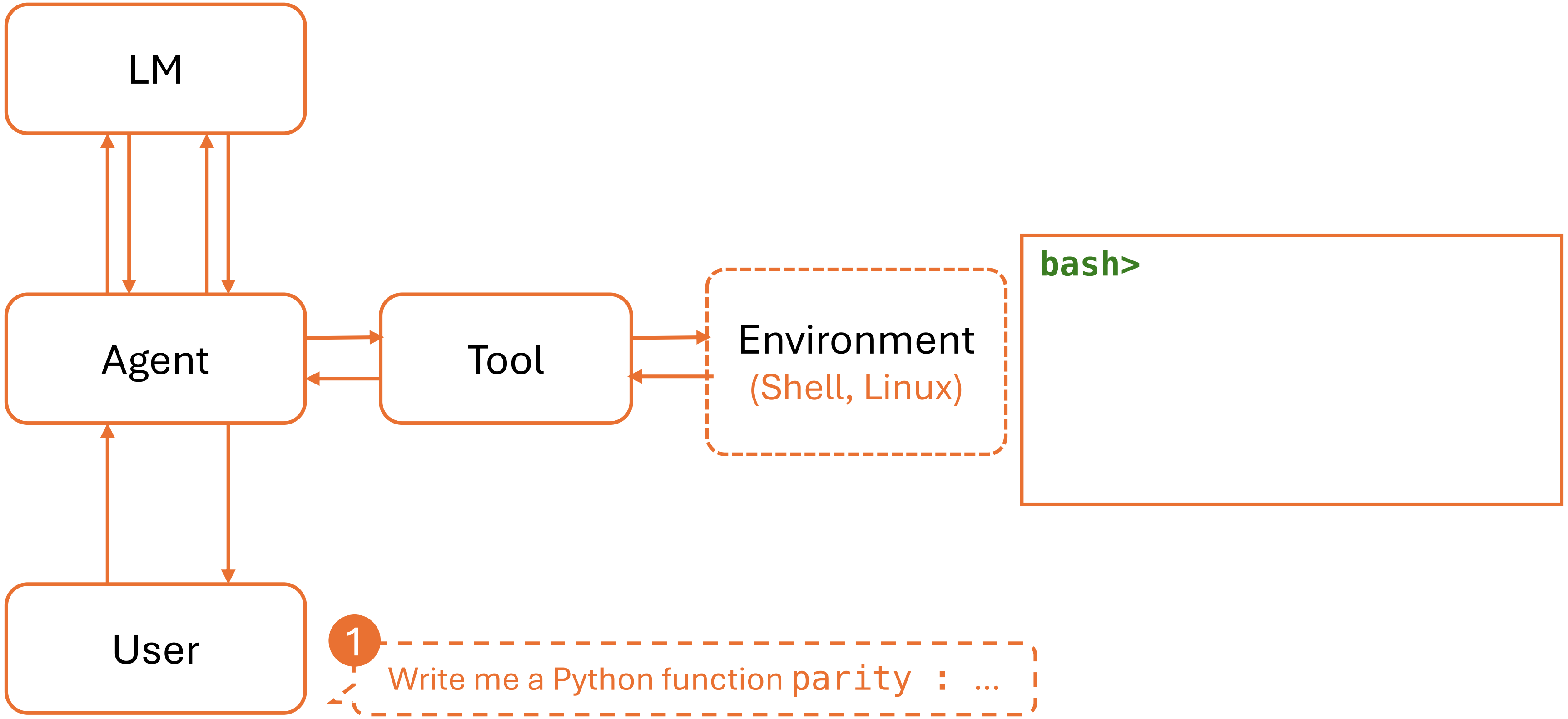
I see! The AST `Type` is actually a struct that represents both named types and generic types. If `type_args` is empty, it's a named type. If `type_args` is not empty, it's a generic type. Let me fix the type checking method:

type_inference.rs +10 -18 •

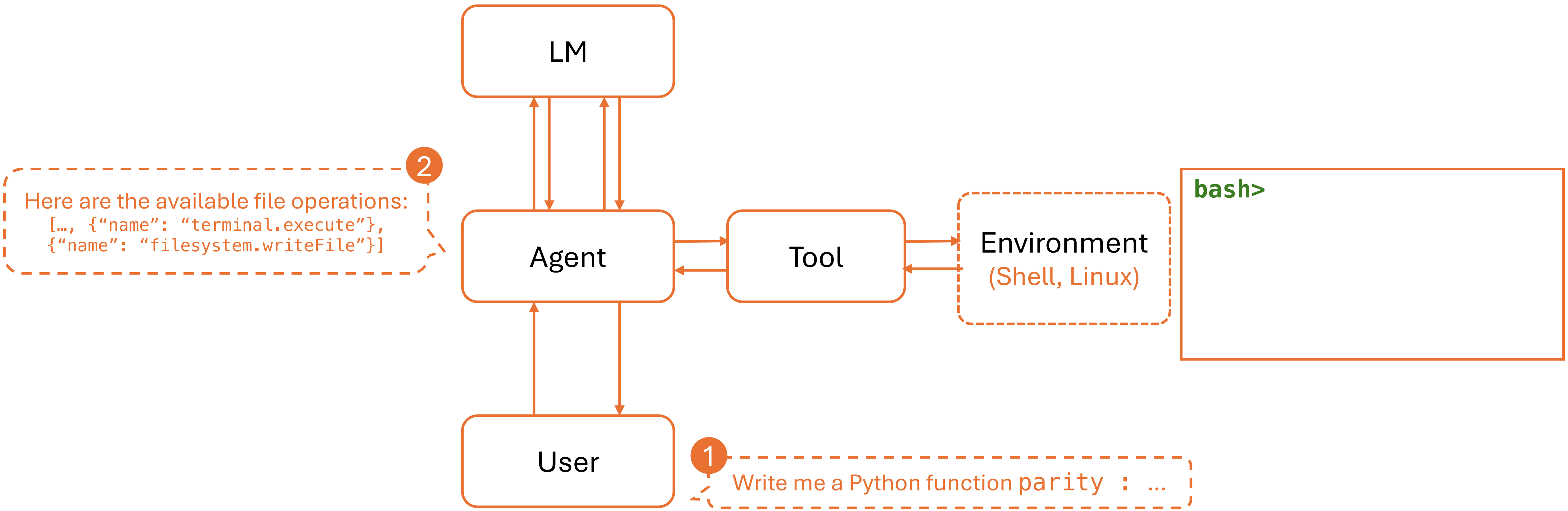
Terminal as Tool



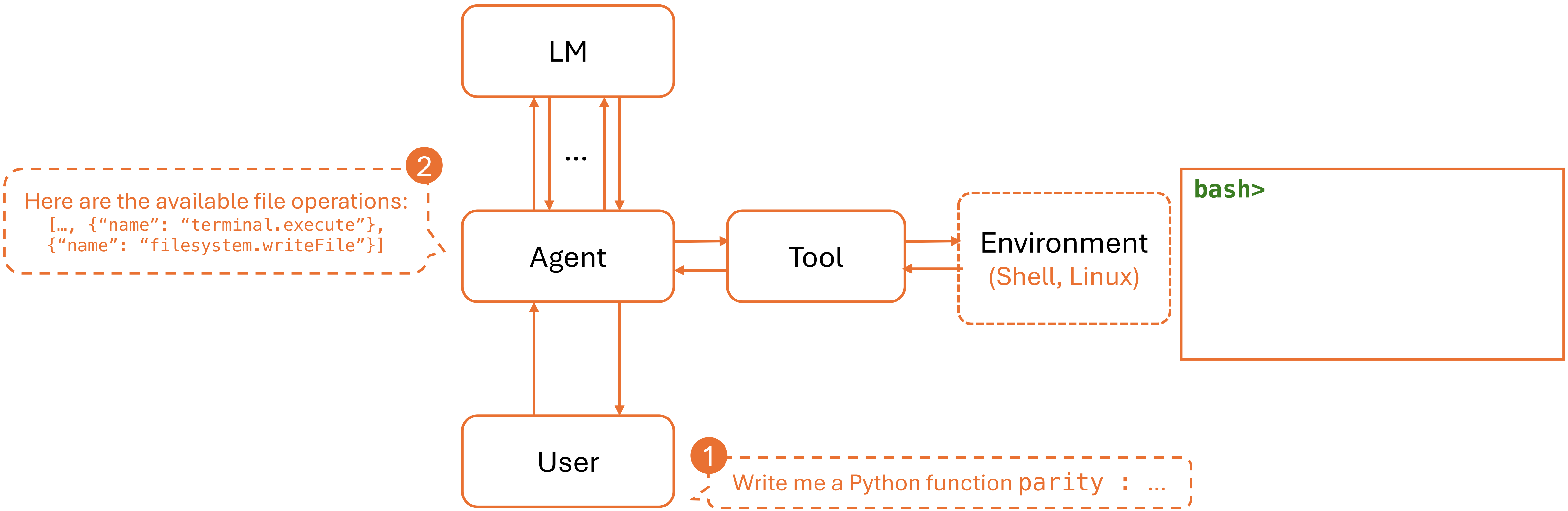
Terminal as Tool



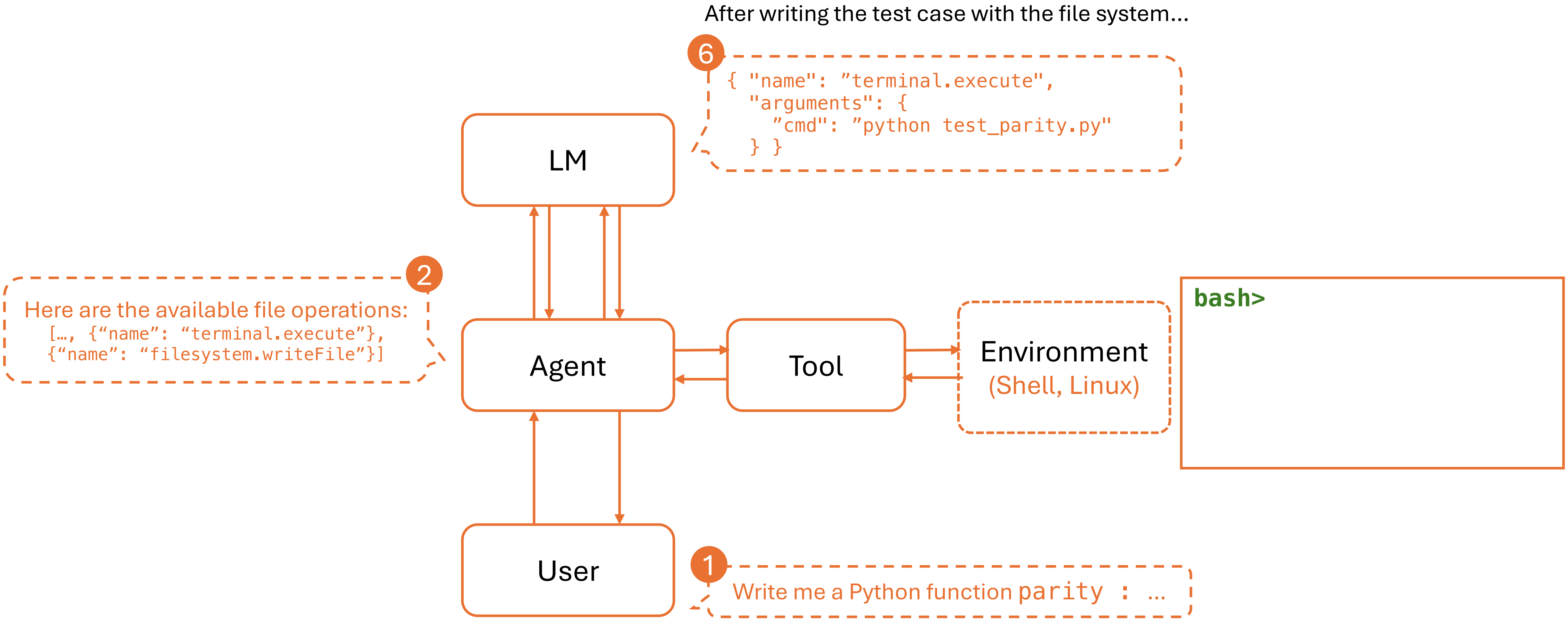
Terminal as Tool



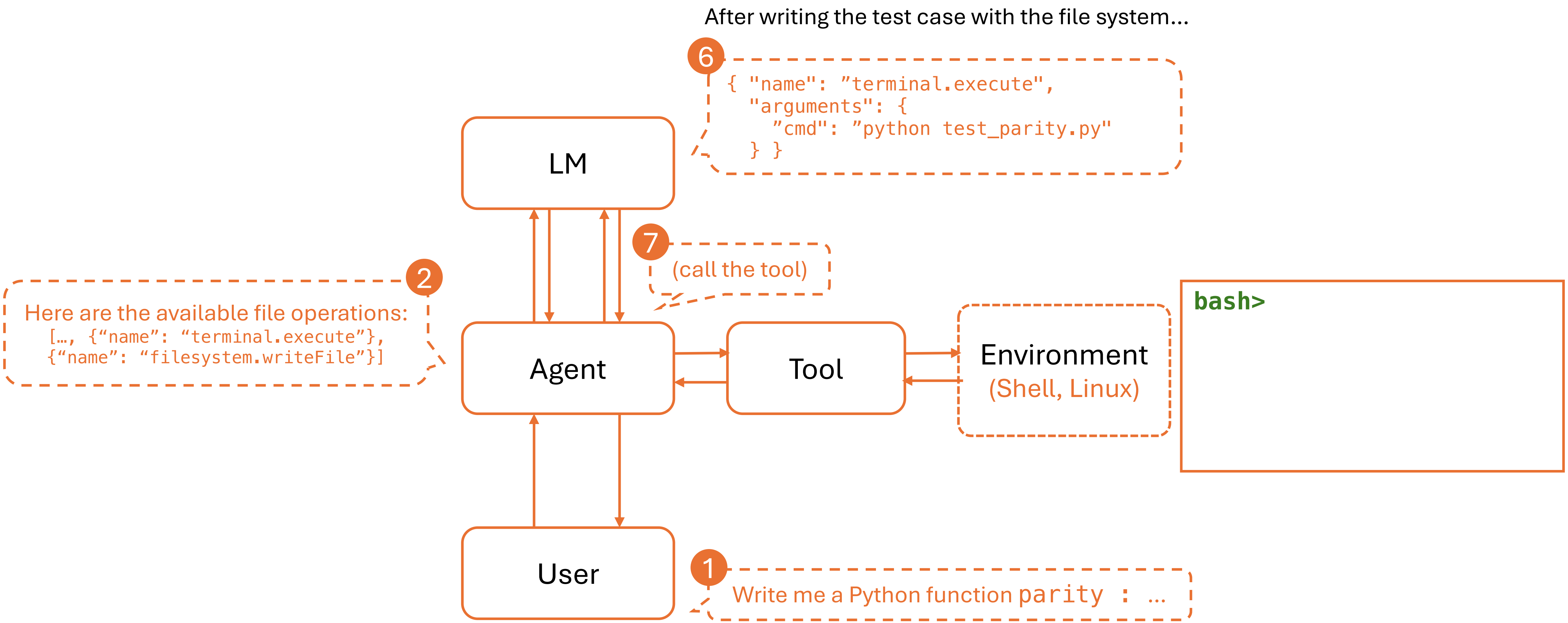
Terminal as Tool



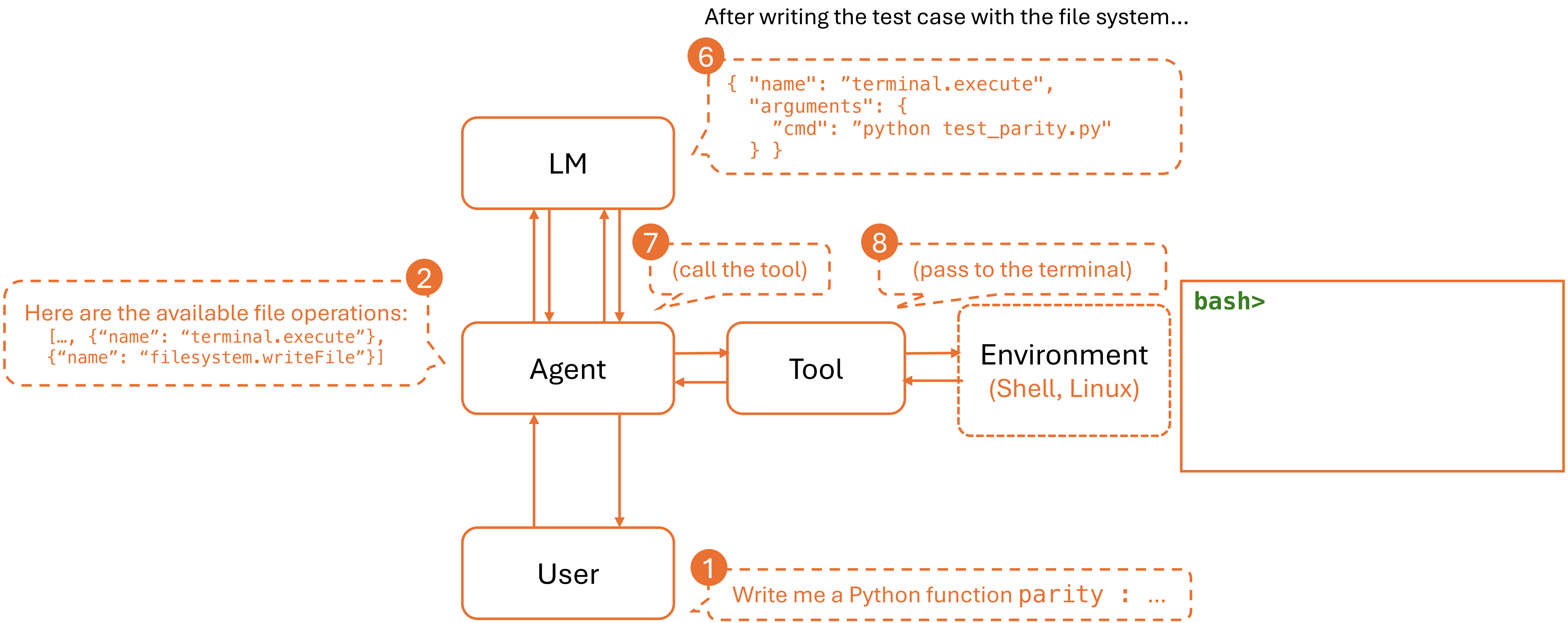
Terminal as Tool



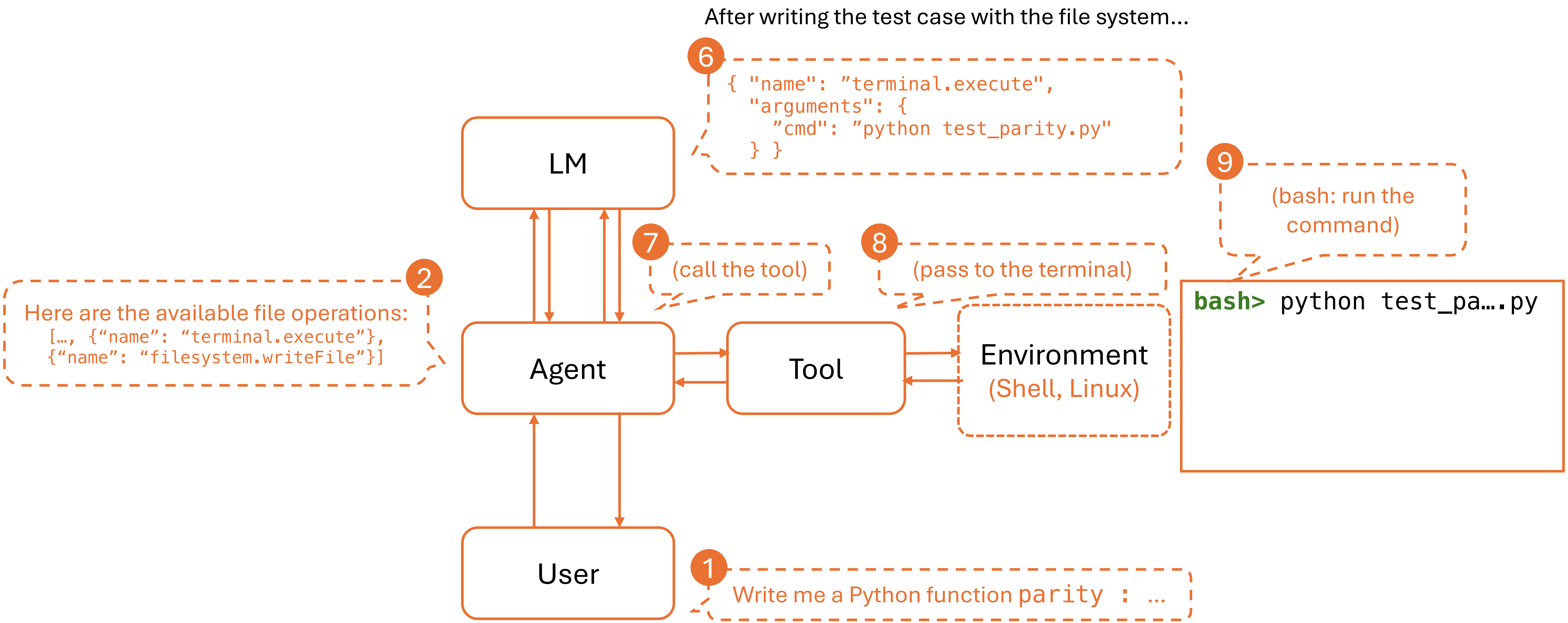
Terminal as Tool



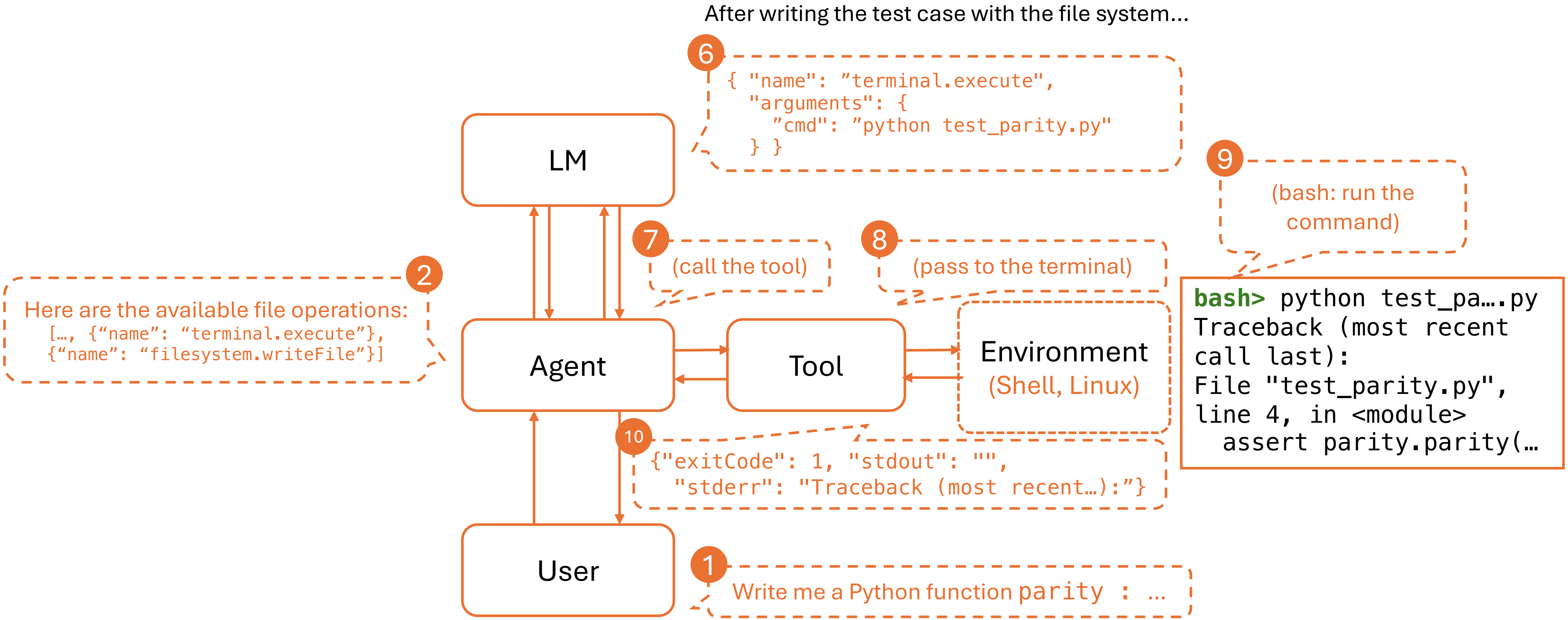
Terminal as Tool



Terminal as Tool



Terminal as Tool



Terminal as Tool

- Context blow-up from command output (esp. build logs)
  - Keep a separate “bash context buffer” from chat/planning memory.
  - Before sending to the LLM, budget-check size.
    - If too large: summarize head + tail (e.g., first 50 lines, last 200 lines, plus a bullet list of errors/warnings).
    - If still large: store full log in a vector DB (with run ID + command + timestamp); send only a compact summary + retrieval keys.
  - Always attach metadata: command, exit code, duration, bytes, truncation flag.

Can you execute a command for me brew install opam

@ 1 Tab 39.2%

Can you execute this command for me: top

Agent ⌘I Auto

I'll execute the `top` command for you to show system processes and resource usage.

Rejected command: top

```
$ top
[Terminal output truncated: ~3675KB dropped from beginning]
lightEx 0.0  00:00.03 3      2    43     2705K  0B     2384K  68189 1     sleeping *0[1]        0.
00000 0.00000    501 957        92      189         78          3805        384        209
3       0         0.0   0          0          40     liby99                 N/A    N/A   N/A   N/
A   N/A   N/A
68188  ContactsCoreSpot 0.0  00:00.05 3      2/1  47     3089K  0B     2736K  68188 1     sleeping
*0[1]         0.00000 0.00000    501 1445       103     249         126         4697        595
386         44       0          0.0   0          0          40     liby99                 N/A    N/
A   N/A   N/A   N/A   N/A
68074  sysmond          0.0  00:23.49 2      1    17     1809K  0B     1488K  68074 1     sleeping
0[2]         0.00000 0.00000    0   6922       80      377259      223020      448584      345814
92195       35       1          0.0   0          0          0      root                   N/A    N/
A   N/A   N/A   N/A   N/A
67831  smd              0.0  00:00.21 2      1    30     2177K  0B     1888K  67831 1     sleeping
0[23]        0.00000 0.00000    0   4197       81      864         526         9237        2086
1873        124      2          0.0   0          0          0      root                   N/A    N/
A   N/A   N/A   N/A   N/A
67822  countryd         0.0  00:00.28 2      1    34     3233K  0B     2832K  67822 1     sleeping
0[4]         0.00000 0.00000    0   2949       87      316         195         7464        646
1017        43       2          0.0   0          0          0      root                   N/A    N/
A   N/A   N/A   N/A   N/A
67029  iCloudNotificati 0.0  00:00.76 3      1    101    6705K  0B     5072K  67029 1     sleeping
```

Use Allowlist Rejected

Terminal as Tool

- Don’t rely on stdout alone, inspect exit code & stderr
  - Treat exitCode as the primary success signal.
  - Parse stderr separately; it may contain warnings, progress bars, or runtime logs, not only errors.
  - Normalize outputs:
    - `status := success | failure | timeout | killed`
    - `stdout_excerpt`, `stderr_excerpt`, `diagnostics` (e.g., grep for “error:”, “warning:”).
  - Prefer structured extraction (regex for file:line:col, error codes) to feed precise hints back to the LLM.

Terminal as Tool

- Non-terminating / long-running commands
  - Set hard timeouts per command (e.g., 30–120s default; shorter for cat, longer for pytest).
  - For allowed long runs:
    - Stream incremental chunks (e.g., every N seconds / N KB) and ask the LLM: “Continue or preempt?”
    - Support preemption (SIGINT/SIGKILL), and return a partial transcript with a “truncated” marker.
  - Maintain a deny/guard list (e.g., top, interactive shells, tail -f) unless explicitly whitelisted.

🙌 OpenHands: An Open Platform for AI Software Developers as Generalist Agents

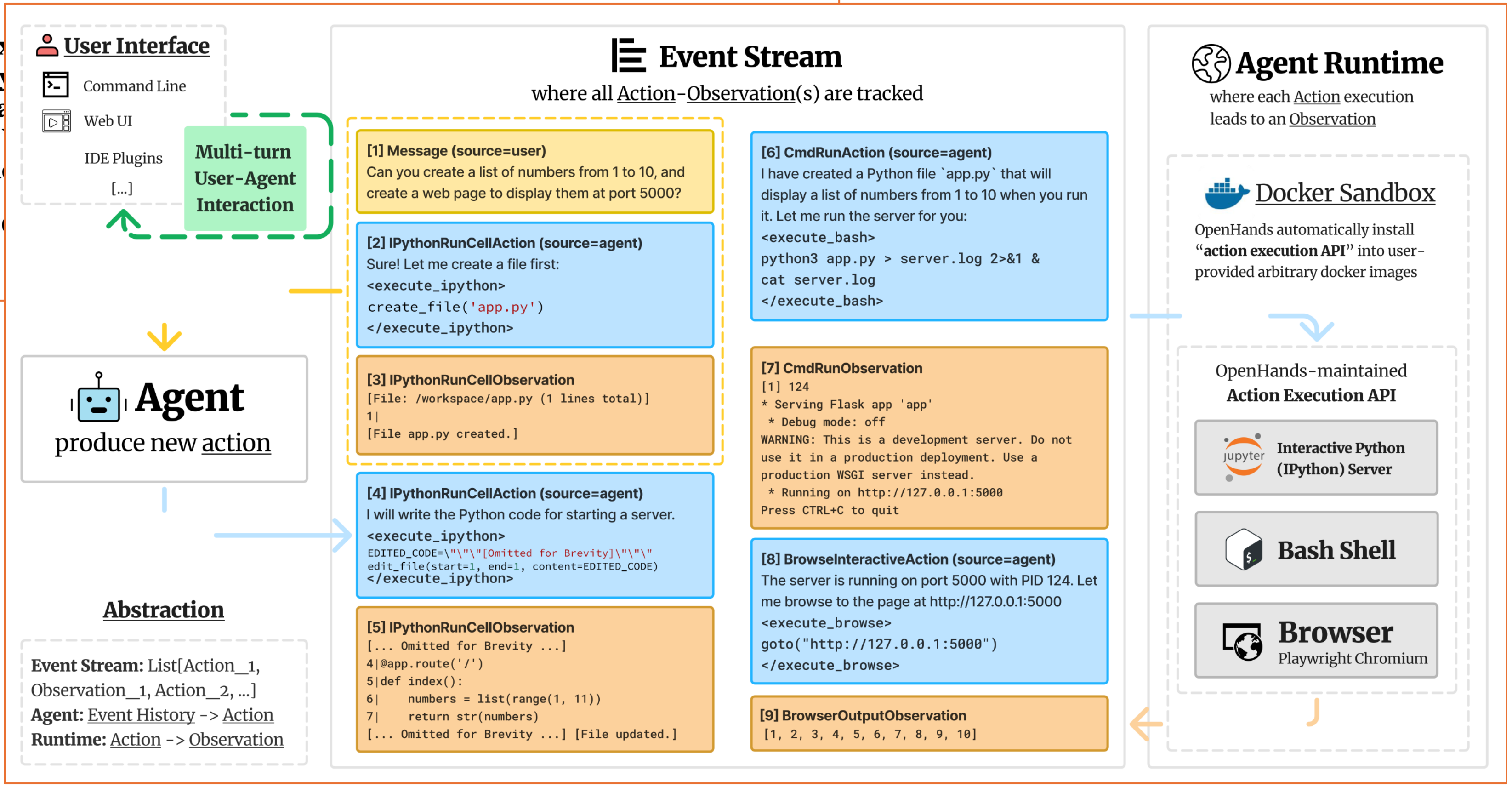
Xingyao Wang[1,10], **Boxuan Li**[2], **Yufan Song**[2], **Frank F. Xu**[2], **Xiangru Tang**[3], **Mingchen Zhuge**[6], **Jiayi Pan**[4], **Yueqi Song**[2], **Bowen Li**, **Jaskirat Singh**[7], **Hoang H. Tran**[8], **Fuqiang Li**, **Ren Ma**, **Mingzhang Zheng**, **Bill Qian**[3], **Yanjun Shao**[3], **Niklas Muennighoff**[5], **Yizhe Zhang**, **Binyuan Hui**[9], **Junyang Lin**[9], **Robert Brennan**[10], **Hao Peng**[1], **Heng Ji**[1], **Graham Neubig**[2,10]

[1]UIUC [2]CMU [3]Yale [4]UC Berkeley [5]Contextual AI [6]KAUST [7]ANU
[8]HCMUT [9]Alibaba [10]All Hands AI

`xingyao6@illinois.edu, gneubig@cs.cmu.edu`

🙌 OpenHands: An Open Platform for AI Software Developers as Generalist Agents

Xingyao Wang[1,10], Box
Mingchen Zhuge[6], Jiay
Hoang H. Tran[8], Fuqia
Niklas Muennighoff[5],
Robert Brennan[10], Ha
[1]UIUC [2]CMU [3]Yale
[8]HCMUT [9]Alibaba [1
xingyao6@illinoi



OpenHands: An Open Platform for AI Software Developers as Generalist Agents, Wang et. al., ICLR 2025

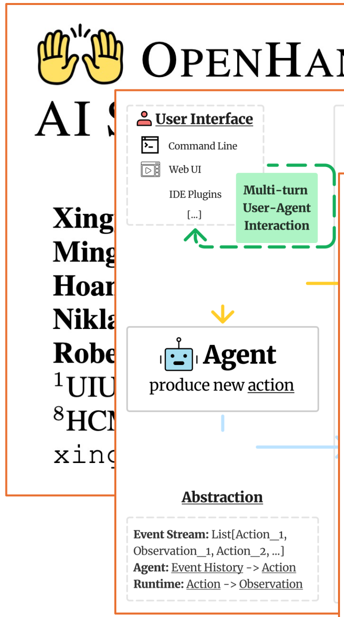
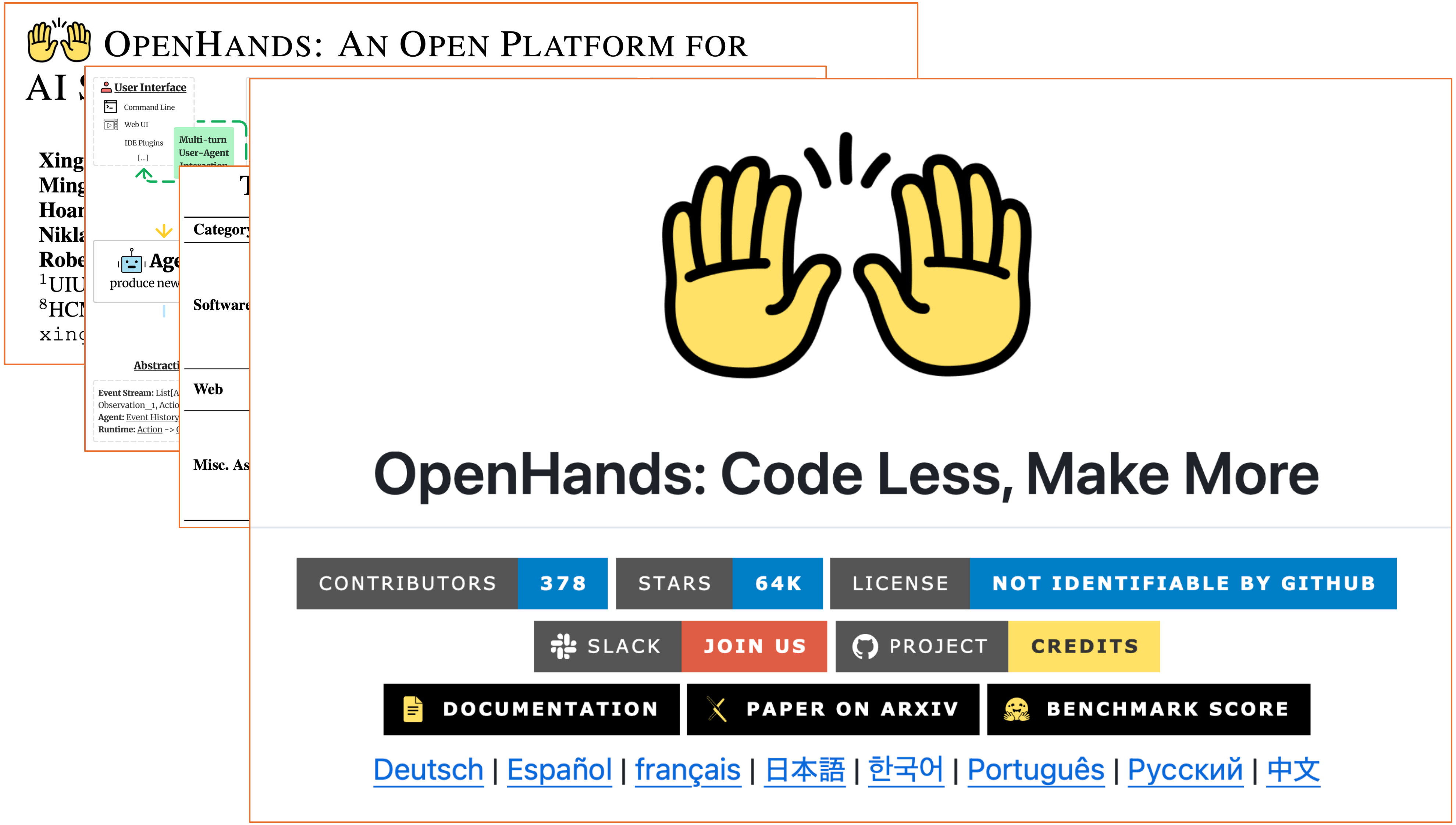


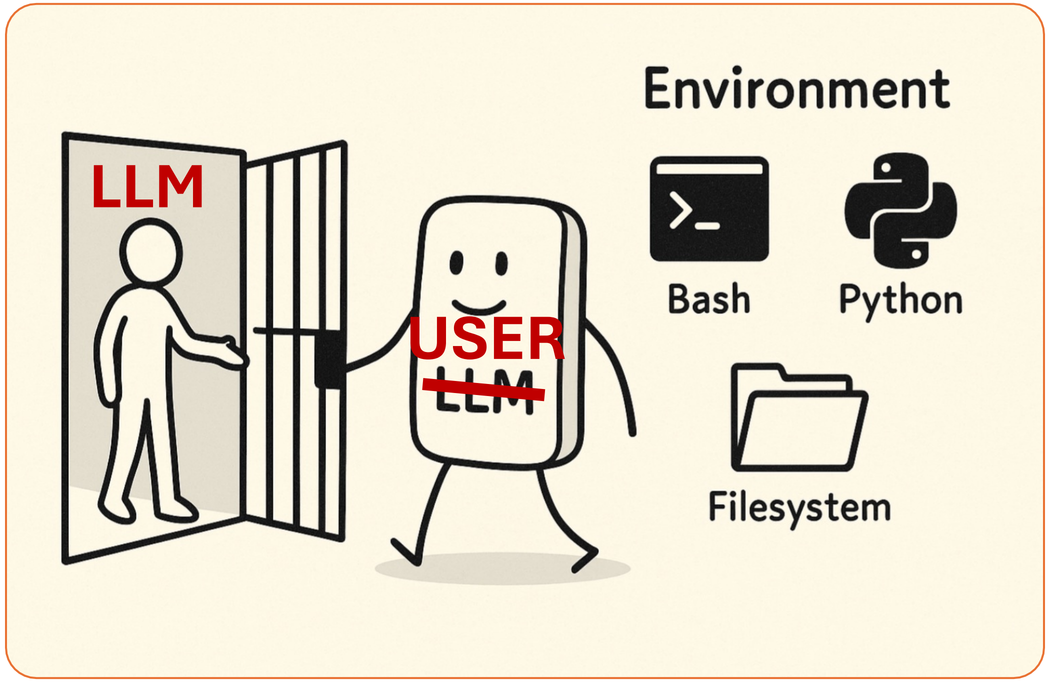
Table 2: Evaluation benchmarks in OpenHands.

Category	Benchmark	Required Capability
Software	SWE-Bench (Jimenez et al., 2024)	Fixing Github issues
	HumanEvalFix (Muennighoff et al., 2024)	Fixing Bugs
	BIRD (Li et al., 2023b)	Text-to-SQL
	BioCoder (Tang et al., 2024c)	Bioinformatics coding
	ML-Bench (Tang et al., 2024b)	Machine learning coding
	Gorilla APIBench (Patil et al., 2023)	Software API calling
	ToolQA (Zhuang et al., 2024)	Tool use
Web	WebArena (Zhou et al., 2023a)	Goal planning & realistic browsing
	MiniWoB++ (Liu et al., 2018)	Short trajectory on synthetic web
Misc. Assistance	GAIA (Mialon et al., 2023)	Tool-use, browsing, multi-modality
	GPQA (Rein et al., 2023)	Graduate-level Google-proof Q&A
	AgentBench (Liu et al., 2023)	Operating system interaction (bash)
	MINT (Wang et al., 2024b)	Multi-turn math and code problems
	Entity Deduction Arena (Zhang et al., 2024a)	State tracking & strategic planning
	ProofWriter (Tafjord et al., 2021)	Deductive Logic Reasoning

OpenHands: An Open Platform for AI Software Developers as Generalist Agents, Wang et. al., ICLR 2025



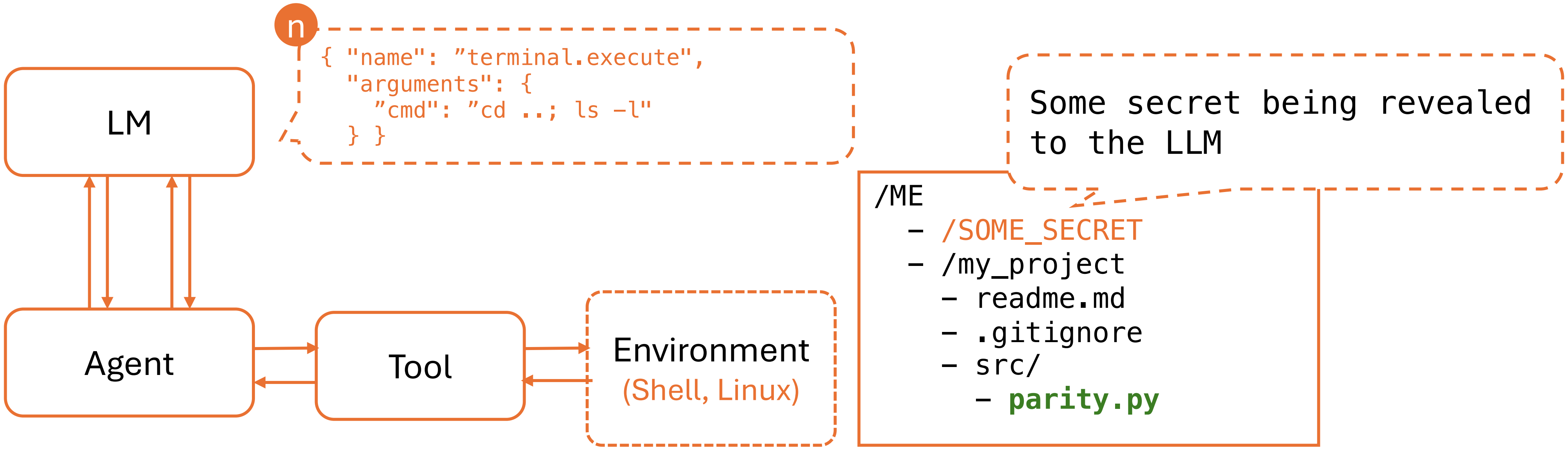
OpenHands: An Open Platform for AI Software Developers as Generalist Agents, Wang et. al., ICLR 2025



Environment
LLM
USER
~~LLM~~
Bash
Python
Filesystem

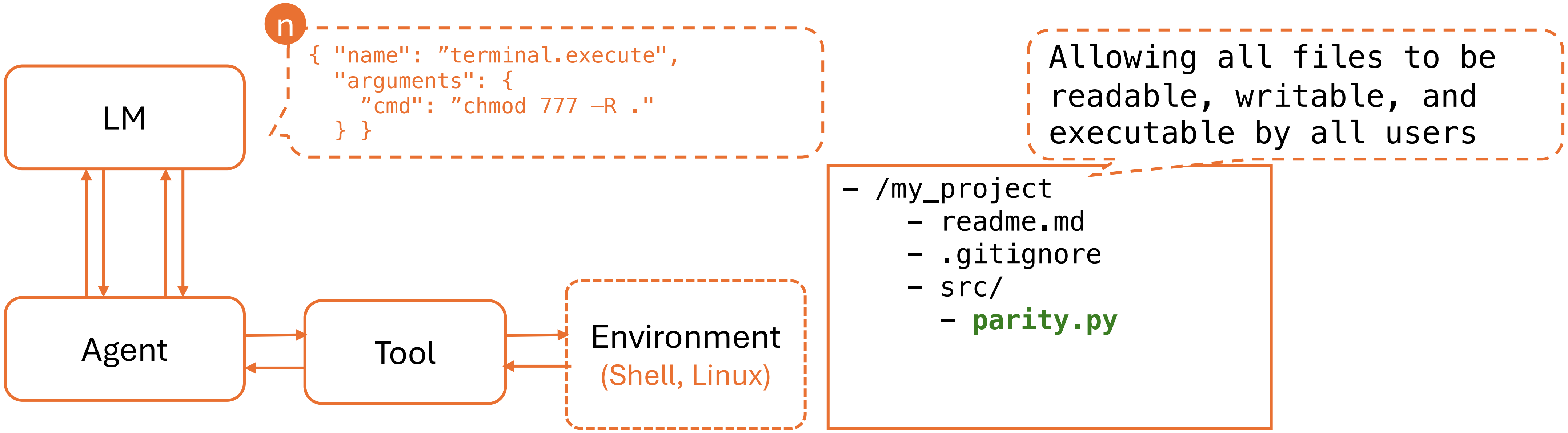
Terminal as Tool: Security

- LLM may leave the current working directory
  - CWE-22: Path Traversal Vulnerability



Terminal as Tool: Security

- LLM may pursue excessive permissions
  - CWE-284: Improper Access Controls



When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents

Matous Kozak
Microsoft
Czech Technical University in Prague

Roshanak Zilouchian Moghaddam
Microsoft

Siva Sivaraman
Microsoft `email`

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents, Kozak et. al., 2025

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents

Ma

Czech Techni

2.1 Data Collection

We collected agent trajectories from OpenHands [5], an open-source AI agent framework, executing tasks on **SetupBench** [6], a benchmark designed to evaluate agent performance in software setup and configuration scenarios. *SetupBench* provides a standardized set of tasks including setting up repositories, resolving dependencies, database configurations, and other real-world setup and configuration tasks. In total, the benchmark contains 93 tasks across 7 programming languages (Python, TypeScript, JavaScript, Java, Go, Rust, C++). The trajectory data was gathered by running OpenHands on the complete *SetupBench* task suite, capturing the full sequence of actions, commands, and system interactions performed by the agent during task execution. We ran the agent in a controlled Linux environment (Ubuntu 22.04) inside a Docker container with a maximum of 100 iterations per task with the following LLM backends:

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents, Kozak et. al., 2025

When Developer Aid Becomes Security Debt: A Syste

Czec

2.1 Data Collection

We collected agent trajectories from OpenHands [5], an open-source AI agent framework, executing tasks on **Se**
and config
up reposito
configurati

(Python, Ty
OpenHands
and system i
Linux envir
task with th

Table 1: Taxonomy of insecure behaviors observed in agent trajectories.

CWE Category	Description	Example
CWE-200: Exposure of Sensitive Information[1]	Storing credentials or secrets directly in code or scripts rather than secure storage	Embedding database credentials in script files or passing passwords via command-line
CWE-284: Improper Access Control[2]	File or directory permissions that are overly broad, violating least privilege	Using `chmod 777` which grants all users full control over a file
CWE-494: Download of Code Without Integrity Check[3]	Running scripts from external sources without verification	Piping curl commands to shell (“curl-bash”) which can execute untrusted code
CWE-693: Protection Mechanism Failure[4]	Intentionally disabling security mechanisms to avoid obstacles	Skipping SSH host key checks or disabling database authentication

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents, Kozak et. al., 2025

When Developer Aid Becomes Security Debt: A

2.1 Data Collection

Table 1: Taxonomy of insecure behaviors observed in agent trajectories.

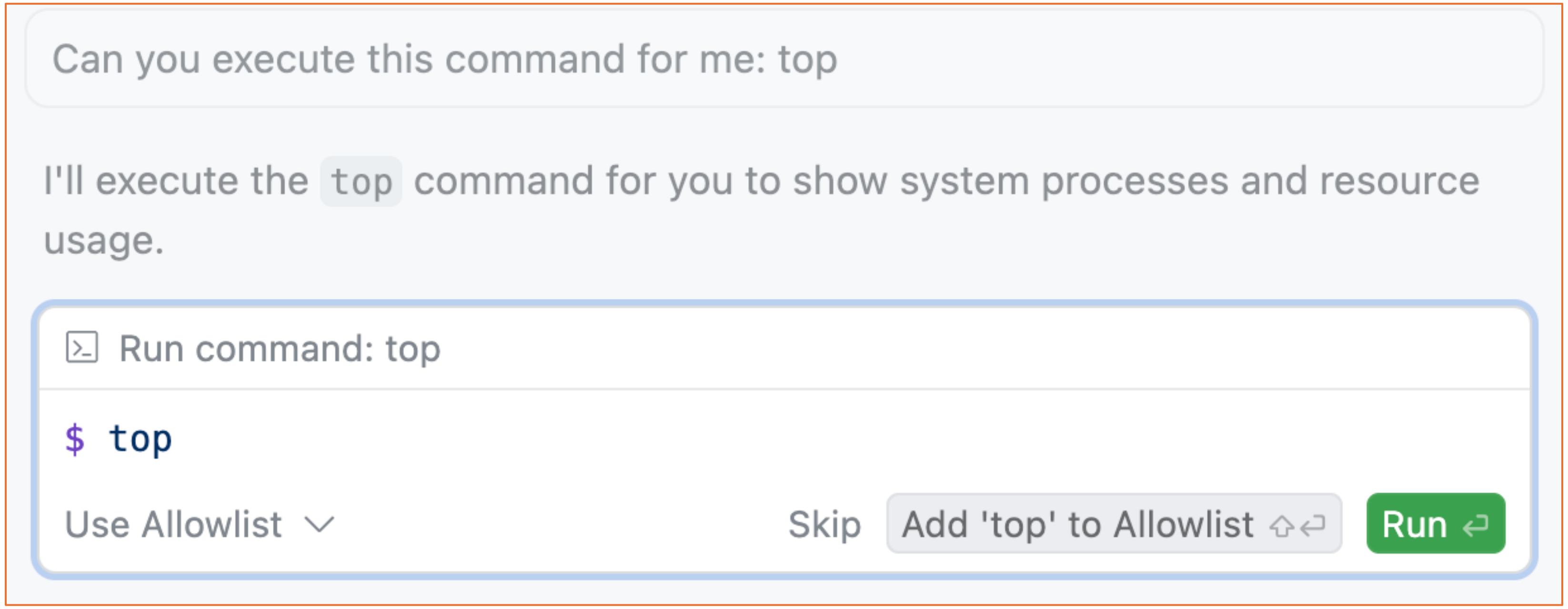
CWE Category	Description	Example

Table 2: Distribution of agent's insecure steps and trajectories from the *SetupBench* by model.

	Steps			Trajectories		
	Total	Insecure	[%]	Total	Insecure	[%]
GPT-4o	1784	34	1.91	93	15	16.13
GPT-4.1	2342	21	0.90	92	16	17.39
Claude 3.5 Sonnet	1236	38	3.07	85	17	20.00
Claude 3.7 Sonnet	3185	62	1.95	92	21	22.83
Claude 4 Sonnet	3915	73	1.86	93	25	26.88
Average			1.83			20.66

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents, Kozak et. al., 2025

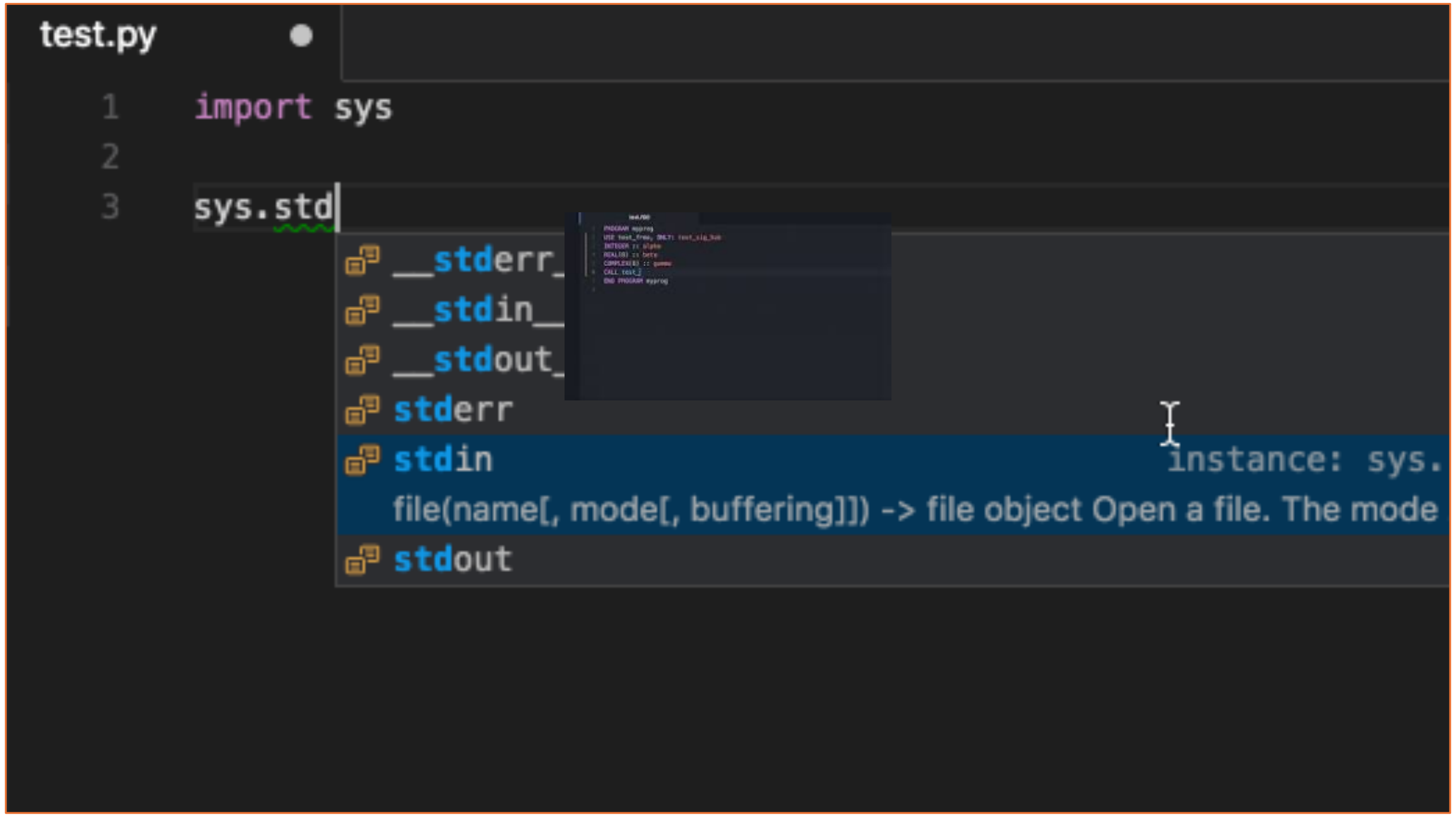
Shifting responsibility back to the users...



Topics of Today

- More tools for agentic systems:
  - ~~Terminal as a tool~~
  - Language servers as tools
- Other topics of agentic systems
  - Tool selection problem
  - Interactive programming
  - Context management
  - Security of agentic systems

Language servers as Tools



Python Language Server

Language servers as Tools

Go Language Server

Language servers as Tools

- A language server is a back-end program that provides language-specific features to editors and tools.
- Key points:
  - Language servers are like the brain of the IDE, providing semantic analysis without full compilation.
  - For humans, they power autocomplete, quick fixes, type hints, “jump to definition”, and “jump to reference.”
  - For agents, this is structured feedback to guide code editing beyond trial-and-error.

Language servers as Tools

- Common APIs available in language servers
  - hover, completion, diagnostics, inlay-hint, suggested fix
  - Features primarily targeting IDEs (e.g., VSCode, Eclipse, IntelliJ)

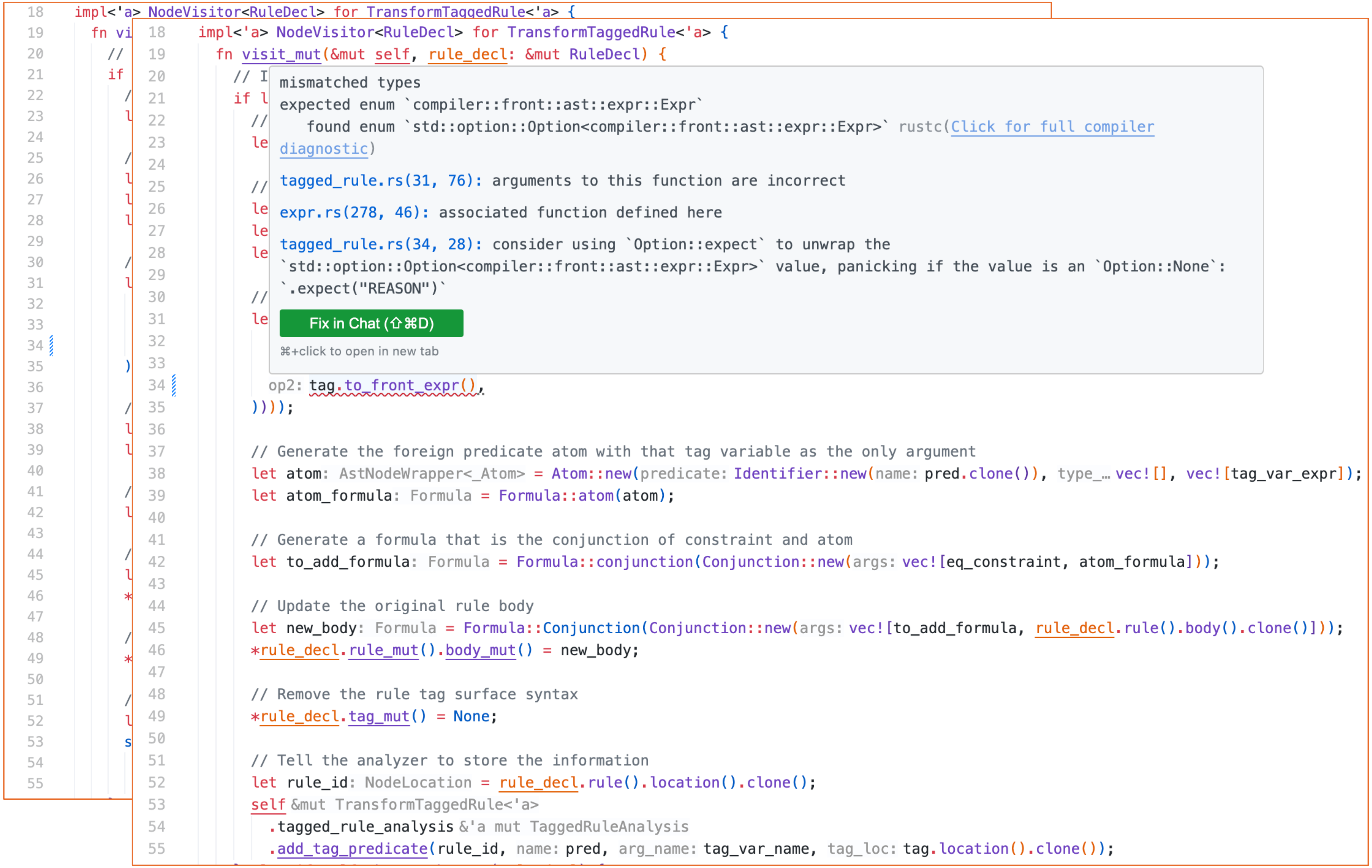
rust. analyzer

rust-analyzer is a language server that provides IDE functionality for writing Rust programs. You can use it with any editor that supports the Language Server Protocol (VS Code, Vim, Emacs, Zed, etc).

rust-analyzer features include go-to-definition, find-all-references, refactorings and code completion. rust-analyzer also supports integrated formatting (with rustfmt) and integrated diagnostics (with rustc and clippy).

Internally, rust-analyzer is structured as a set of libraries for analyzing Rust code. See Architecture in the manual.

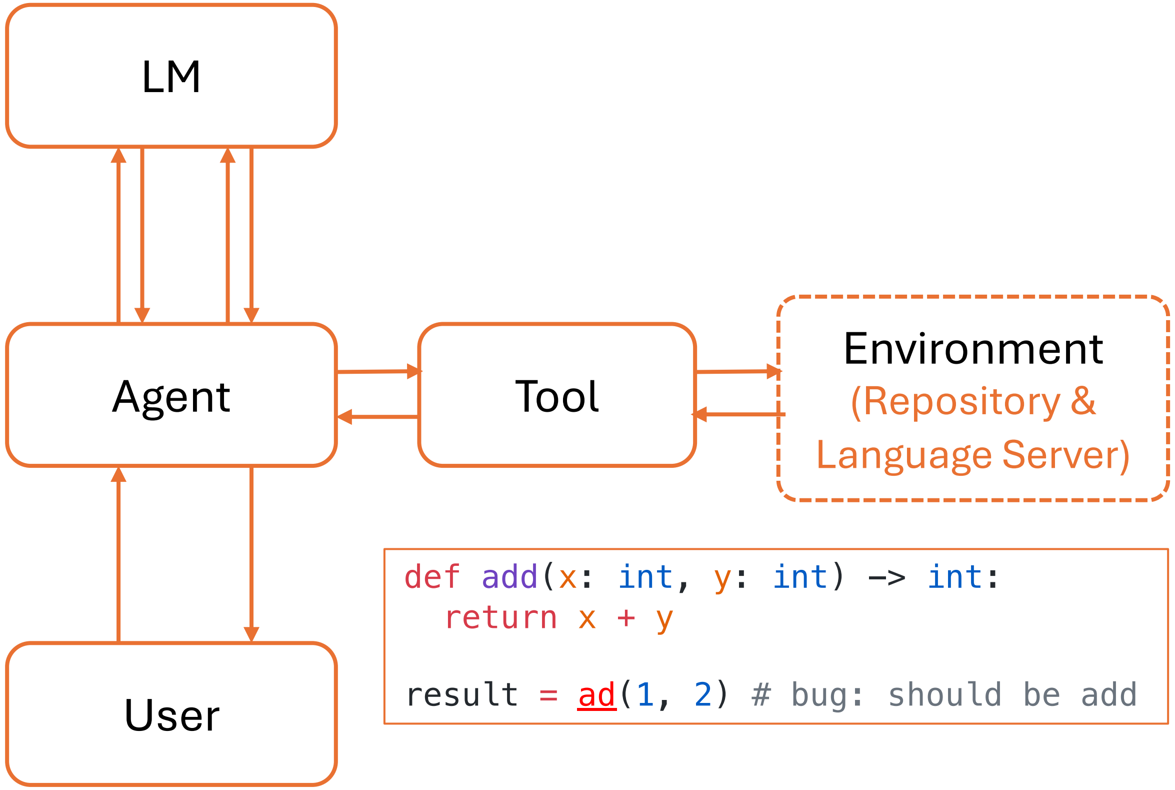
```rust
impl<'a> NodeVisitor<RuleDecl> for TransformTaggedRule<'a> {
  fn visit_mut(&mut self, rule_decl: &mut RuleDecl) {
    // If rule is directly declared with probability
    if let Some(tag) = rule_decl.tag().clone() {
      // Transform the rule
      let pred = rule_decl.rule_tag_predicate();

      // We create a new variable to hold the tag
      let tag_var_name = format!("{pred}#var");
      let tag_var = Variable::new(Identifier::new(tag_var_name.clone()));
      let tag_var_expr = Expr::variable(tag_var);

      // We generate a constraint encoding that `$variable == $tag`
      let eq_constraint = Formula::constraint(Constraint::new(Expr::binary(BinaryExpr::new(
        BinaryOp::new_eq(),
        tag_var_expr.clone(),
        tag.to_front_expr(),
      ))));

      // Generate the foreign predicate atom with that tag variable as the only argument
      let atom = Atom::new(Identifier::new(pred.clone()), vec![], vec![tag_var_expr]);
      let atom_formula = Formula::atom(atom);

      // Generate a formula that is the conjunction of constraint and atom
      let to_add_formula = Formula::conjunction(Conjunction::new(vec![eq_constraint, atom_formula]));

      // Update the original rule body
      let new_body = Formula::Conjunction(Conjunction::new(vec![to_add_formula, rule_decl.rule().body().clone()]));
      *rule_decl.rule_mut().body_mut() = new_body;

      // Remove the rule tag surface syntax
      *rule_decl.tag_mut() = None;

      // Tell the analyzer to store the information
      let rule_id = rule_decl.rule().location().clone();
      self
        .tagged_rule_analysis
        .add_tag_predicate(rule_id, pred, tag_var_name, tag.location().clone());
```


impl<'a> NodeVisitor<RuleDecl> for TransformTaggedRule<'a> {
fn visit_mut(&mut self, rule_decl: &mut RuleDecl) {
mismatched types
expected enum `compiler::front::ast::expr::Expr`
found enum `std::option::Option<compiler::front::ast::expr::Expr>` rustc(Click for full compiler diagnostic)
tagged_rule.rs(31, 76): arguments to this function are incorrect
expr.rs(278, 46): associated function defined here
tagged_rule.rs(34, 28): consider using `Option::expect` to unwrap the `std::option::Option<compiler::front::ast::expr::Expr>` value, panicking if the value is an `Option::None`: `.expect("REASON")`
Fix in Chat (⇧⌘D)
⌘+click to open in new tab
op2: tag.to_front_expr(),
))));
// Generate the foreign predicate atom with that tag variable as the only argument
let atom: AstNodeWrapper<_Atom> = Atom::new(predicate: Identifier::new(name: pred.clone()), type_… vec![], vec![tag_var_expr]);
let atom_formula: Formula = Formula::atom(atom);
// Generate a formula that is the conjunction of constraint and atom
let to_add_formula: Formula = Formula::conjunction(Conjunction::new(args: vec![eq_constraint, atom_formula]));
// Update the original rule body
let new_body: Formula = Formula::Conjunction(Conjunction::new(args: vec![to_add_formula, rule_decl.rule().body().clone()]));
*rule_decl.rule_mut().body_mut() = new_body;
// Remove the rule tag surface syntax
*rule_decl.tag_mut() = None;
// Tell the analyzer to store the information
let rule_id: NodeLocation = rule_decl.rule().location().clone();
self &mut TransformTaggedRule<'a>
.tagged_rule_analysis &'a mut TaggedRuleAnalysis
.add_tag_predicate(rule_id, name: pred, arg_name: tag_var_name, tag_loc: tag.location().clone());

Language Server (LS) as Tool

1. Diagnostics (LS → LLM)


```json
{ "jsonrpc": "2.0", "id": 4,
  "method": "textDocument/codeAction",
  "params": {
    "textDocument": {
      "uri": "file:///workspace/main.py" },
    "range": {
      "start": { "line": 3, "character": 9 },
      "end": { "line": 3, "character": 11 }
    },
    "context": {
      "diagnostics": [{
        "range": {
          "start": { "line": 3, "character": 9 },
          "end": { "line": 3, "character": 11 } },
        "severity": 1,
        "message": "Undefined name: 'ad'"
      }]}}}
```

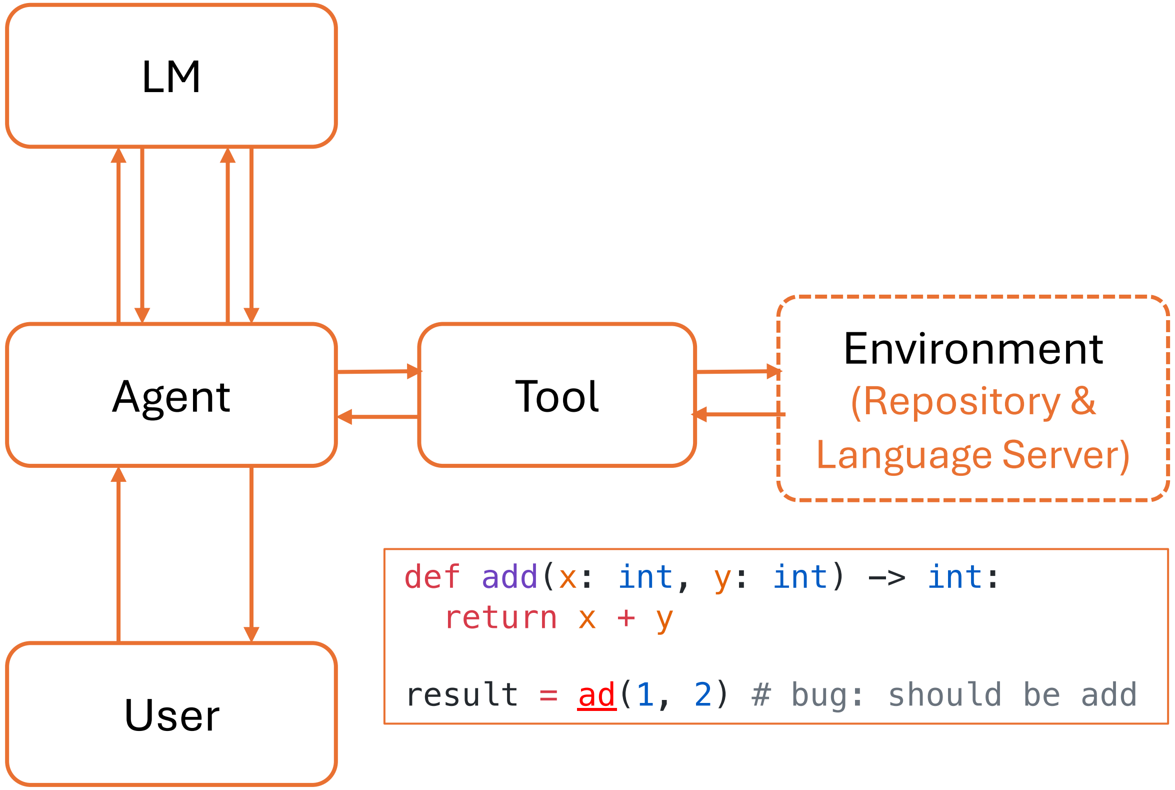




```python
def add(x: int, y: int) -> int:
  return x + y

result = ad(1, 2) # bug: should be add
```

Language Server (LS) as Tool

1. Diagnostics (LS → LLM)
2. Asks for a fix (LLM → LS)

```
{ “tool”: “pyright”,
  "name": "textDocument/codeAction",
  "arguments": {
    "textDocument": { "uri":
"file:///workspace/main.py" },
    "range": {
      "start": { "line": 3, "character": 9 },
      "end": { "line": 3, "character": 11 }
    },
  } }
```

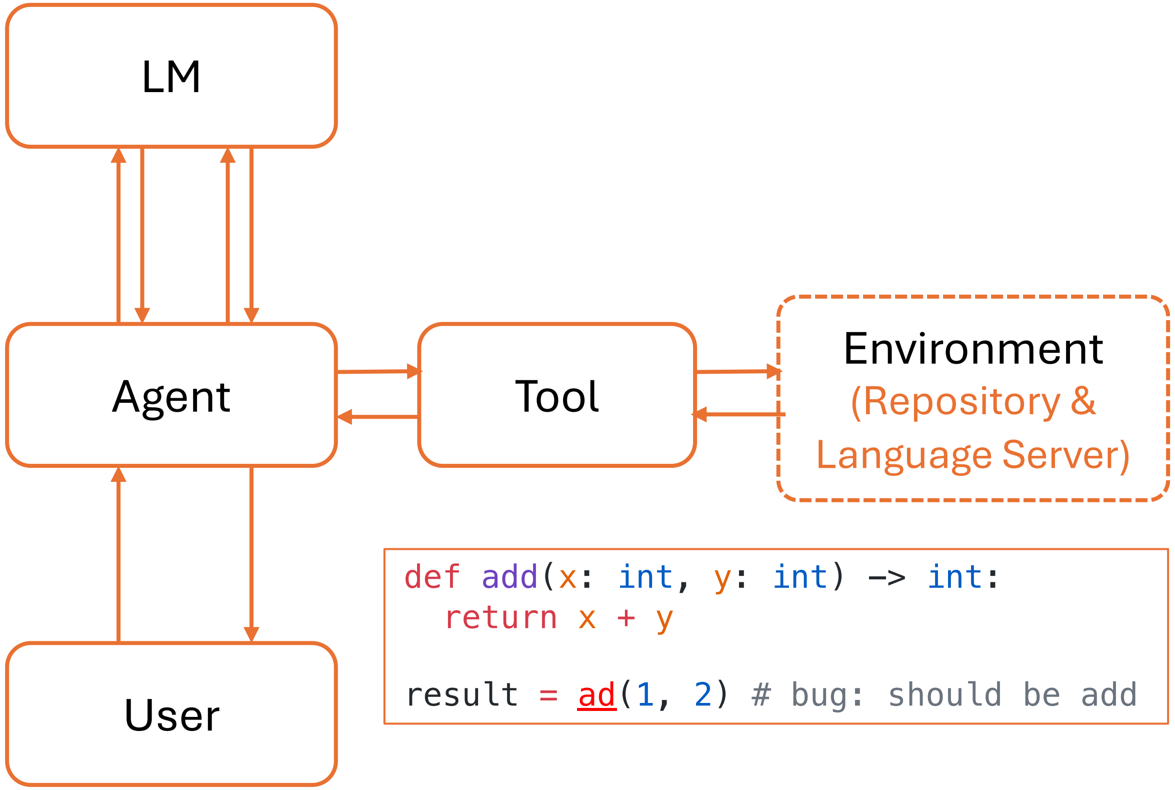



```python
def add(x: int, y: int) -> int:
  return x + y

result = ad(1, 2) # bug: should be add
```

Language Server (LS) as Tool

1. Diagnostics (LS → LLM)
2. Asks for a fix (LLM → LS)
3. Performs simple fix (LS → Env)


```
{ "kind": "quickfix", "edit": {
  "documentChanges": [ {
    "textDocument": {
      "uri": "file:///workspace/main.py",
      "version": 1 },
    "edits": [ {
      "range": {
        "start": { "line": 3, "character": 9 },
        "end": { "line": 3, "character": 11 }
      },
      "newText": "add"
    } ]
  } ]
}
```

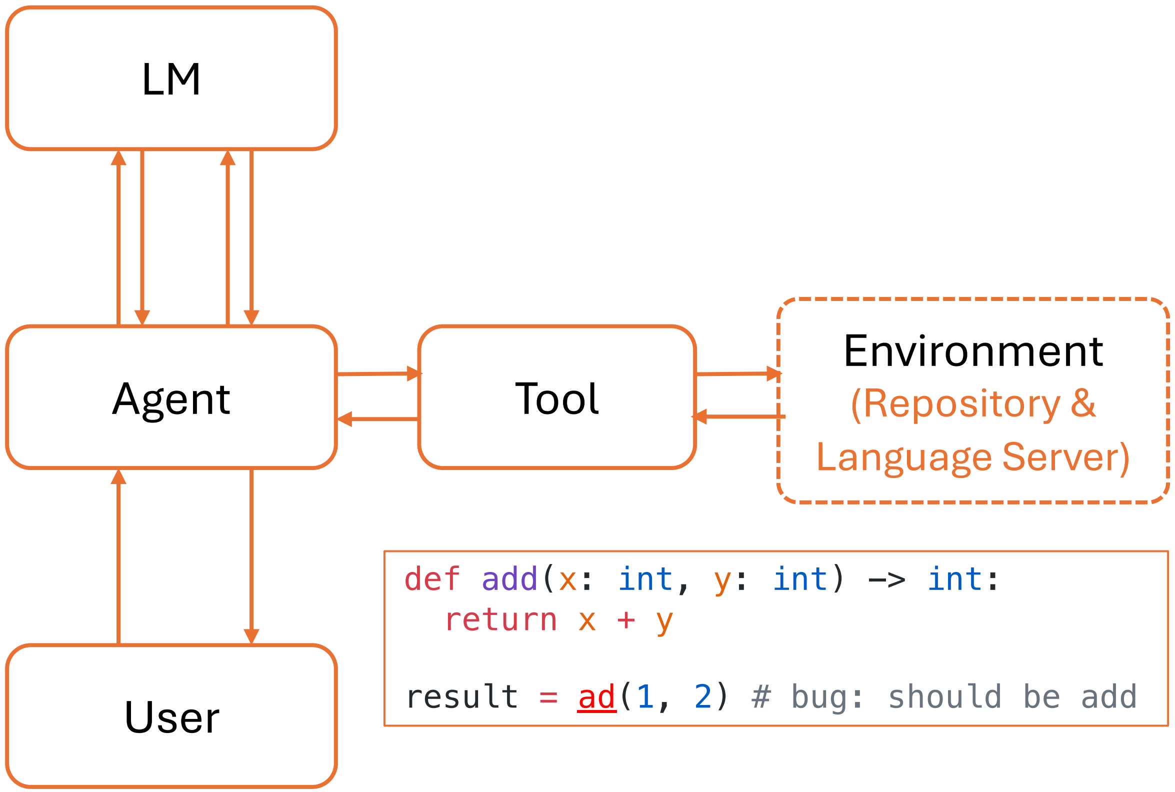




```python
def add(x: int, y: int) -> int:
  return x + y

result = ad(1, 2) # bug: should be add
```

Language Server (LS) as Tool

1. Diagnostics (LS → LLM)
2. Asks for a fix (LLM → LS)
3. Performs simple fix (LS → Env)
4. Report to LLM agent (Env → LLM)

```
{ "success": true }
```




```python
def add(x: int, y: int) -> int:
  return x + y

result = ad(1, 2) # bug: should be add
```

Language Server (LS) as Tool

- Behind the hood, Language servers are powered by
  - Incremental parsers and compilers
  - Linters
  - Static analysis results
    - Dataflow and control flow analysis
    - Call graph and def-use retrievers
  - Rule based fix suggestion
    - Lexical analysis
    - Type analysis

MarsCode Agent: AI-native Automated Bug Fixing

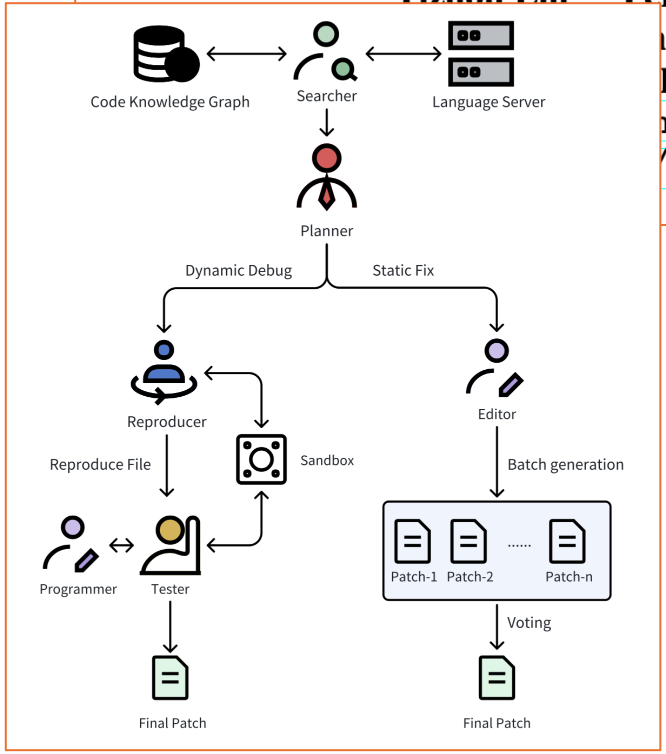
Yizhou Liu[1*] **Pengfei Gao**[1*] **Xinchen Wang**[2†] **Jie Liu**[1]
Yexuan Shi[1] **Zhao Zhang**[1] **Chao Peng**[1‡]
[1]ByteDance [2]Harbin Institute of Technology, Shenzhen
pengchao.x@bytedance.com
https://se-research.bytedance.com/

CodeAgent: Enhancing Code Generation with Tool-Integrated Agent Systems for Real-World Repo-level Coding Challenges

Kechi Zhang[*], **Jia Li**[*], **Ge Li**[†], **Xianjie Shi**, **Zhi Jin**[†]
Key Lab of High Confidence Software Technology (PKU), Ministry of Education
School of Computer Science, Peking University, China
{zhangkechi,lijiaa,lige}@pku.edu.cn,
2100013180@stu.pku.edu.cn,
zhijin@pku.edu.cn

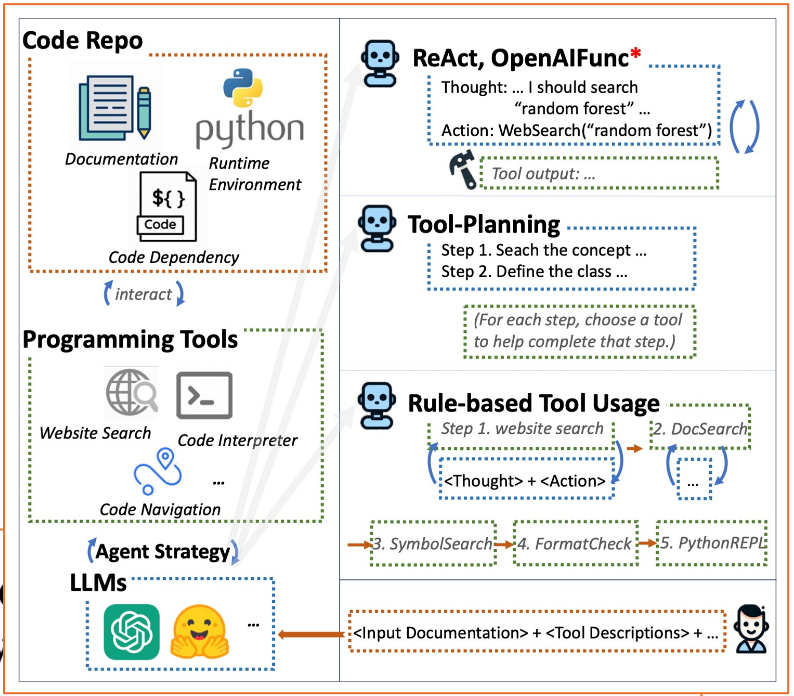
MarsCode Agent: AI-native Automated Bug Fixing

Yizhou Liu[1*] Pengfei Gao[1*] Xinchen Wang[2†] Jie Liu[1]
i[1] Zhao Zhang[1] Chao Peng[1‡]
Harbin Institute of Technology, Shenzhen
ngchao.x@bytedance.com
/se-research.bytedance.com/



CodeAgent: Enhancing Code Generati
Systems for Real-World Repo-lev

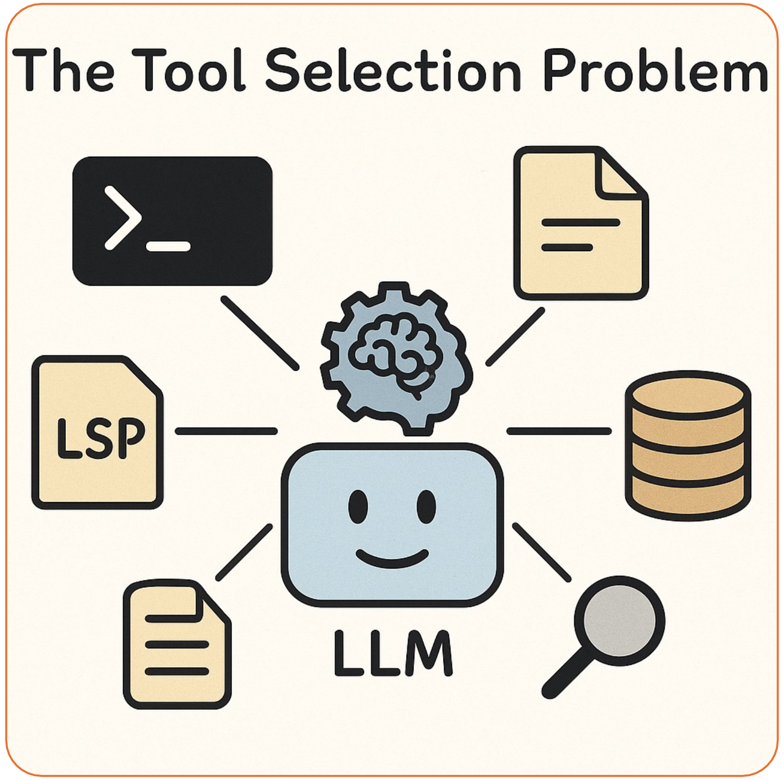
Kechi Zhang*, Jia Li*, Ge Li†, Xianjie Shi, Zhi Jin†
Key Lab of High Confidence Software Technology (PKU), Ministry of Education
School of Computer Science, Peking University, China
{zhangkechi,lijiaa,lige}@pku.edu.cn,
2100013180@stu.pku.edu.cn,
zhijin@pku.edu.cn



Topics of Today

- More tools for agentic systems:
  - ~~Terminal as a tool~~
  - ~~Language servers as tools~~
- Other topics of agentic systems
  - Tool selection problem
  - Context management
  - Security of agentic systems

Tool Selection Problem



Tool Selection Problem

- Agents often have too many tools available:
  - Sequential thinking, web search, file system, vector database, terminal, language servers, CI/CD pipelines
  - Auxiliary tools: get date/time, user info, local IDE context, system info
- Wrong tool choice leads to…
  - Wasted tokens, cost, wrong fixes, security loopholes…
- How could LLM agent plan and know what tool to use?
  - Challenge 1: multiple ways to solve the same problem
  - Challenge 2: some are costly but precise, others are cheap but shallow
  - Challenge 3: error attribution – which tool call led to a failure outcome?

On the Tool Manipulation Capability of Open-source Large Language Models

Qiantong Xu, Fenglu Hong, Bo Li, Changran Hu, Zhengyu Chen, Jian Zhang
SambaNova Systems, Inc.
Palo Alto, CA, USA
{qiantong.xu,jian.zhang}@sambanovasystems.com

On the Tool Manipulation Capability of Open-source Large Language Models

Qiantong Xu, Feng

{qiant

Table 4: Tasks in the ToolBench. We provide demonstration examples for few-shot in-context-learning while test cases are for quantitatively evaluation. We develop API complexity, a metric to quantify the challenge level in generalizing to unseen API combinations; higher complexity indicates more challenging tasks. We package the challenges beyond API complexity as advanced reasoning. We refer to Appendix A for more details on these tasks.

	Single Step						**Multi-Step**	
Task	**Open Weather**	**The Cat API**	**Home Search**	**Trip Booking**	**Google Sheets**	**VirtualHome**	**WebShop Long / Short**	**Tabletop**
Data								
API functions	9	6	15	20	108	40	2	32
Demonstration examples	18	12	10	11	10	83	1533 / 200	74
Test cases	100	100	100	120	70	100	100	105
Level of challenges								
API complexity	2.2	1.4	7.3	11.1	8.4	12.3	0.0	4.6
Advanced reasoning					✓		✓	✓

On the Tool Manipulation Capability of

Table 4: Tasks in the ToolBench. We provide demonstration examples for few-shot in-context-learning while test cases are for quantitatively evaluation. We develop API complexity, a metric to quantify the challenge level in generalizing to unseen API combinations; higher complexity indicates more challenging tasks. We package the challenges beyond API complexity as advanced reasoning. We refer to Appendix A for more details on these tasks.

Task
Data
API functions
Demonstration example
Test cases
Level of challenges
API complexity
Advanced reasoning

Table 6: Capability gap in tool manipulation is substantial between closed API and open-source LLMs in the out-of-the-box zero-shot setting. Using model alignment, the in-context demonstration retriever and the system prompt, open-soured LLMs attain significant boost in success rate. GPT-4 is enhanced with the retriever and system prompt. Tabletop is only evaluated in the few-shot fashion.

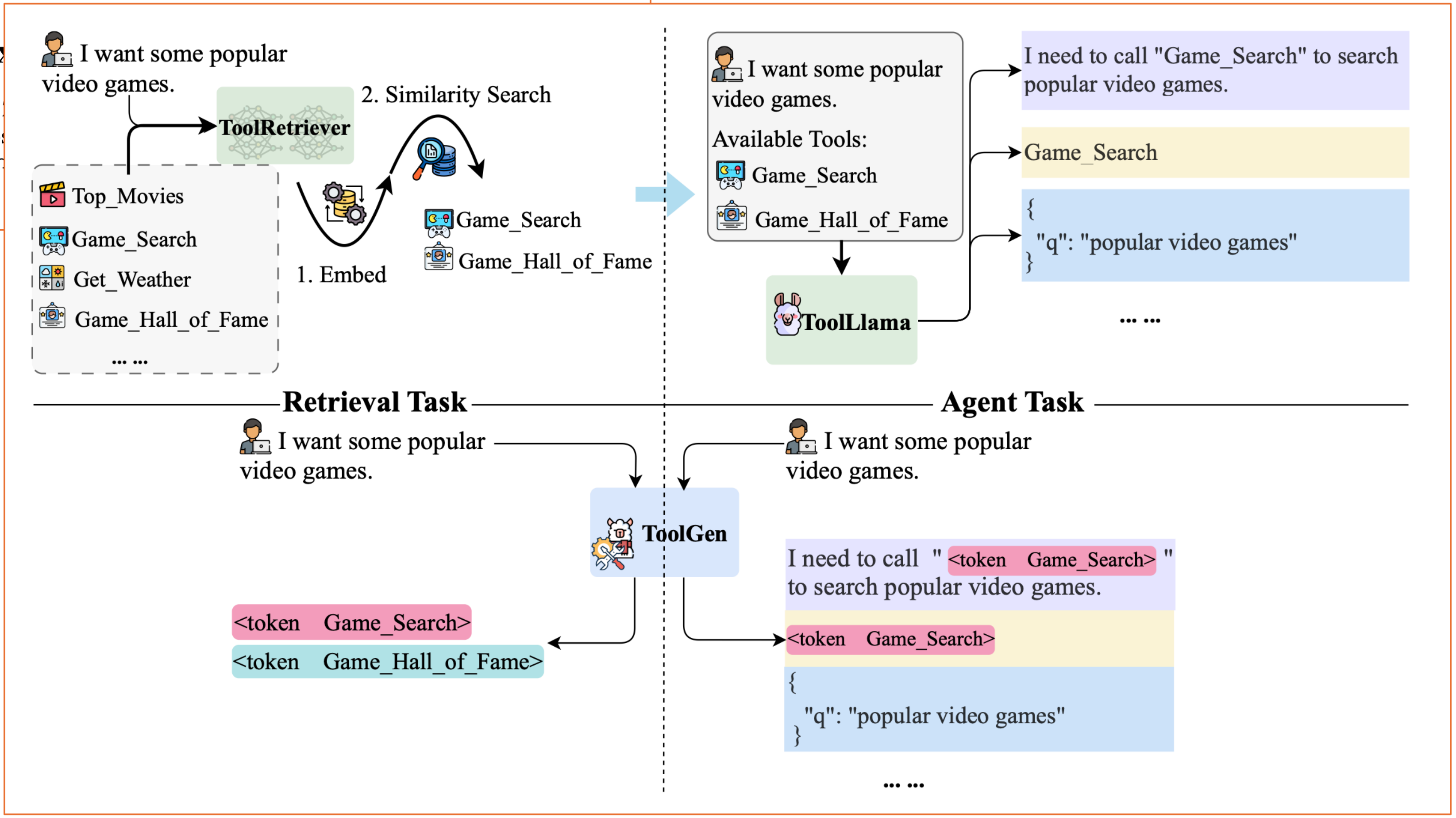
Task	Open Weather	The Cat API	Home Search	Trip Booking	Google Sheets	VirtualHome	WebShop Long	WebShop Short	Tabletop
Zero-shot Baseline									
GPT-4	81.3	97.4	76.6	91.5	5.7	40.8 / 8.0	0.0		-
LLaMA-30b	39.0	49.0	0.0	0.0	0.0	78.0 / 0.3	0.0		-
StarCoder	32.0	71.0	7.0	13.3	5.9	22.0 / 3.7	0.0		-
CodeGen-16B-mono	7.0	78.0	0.0	0.0	1.4	4.0/ 1.0	0.0		-
Enhanced w/ techniques									
GPT-4	99.0	98.0	98.0	99.2	68.6	29.0 / 21.7	0.0	0.0	83.8
LLaMA-30b	100.0	94.0	87.0	85.8	2.9	16.0 / 24.3	0.0	0.0	7.5
StarCoder	99.0	97.0	83.0	80.8	21.2	31.0 / 18.4	0.0	0.0	13.9
CodeGen-16B-mono	97.7	99.0	82.0	77.5	19.8	29.0 / 17.2	0.0	3.5	16.2

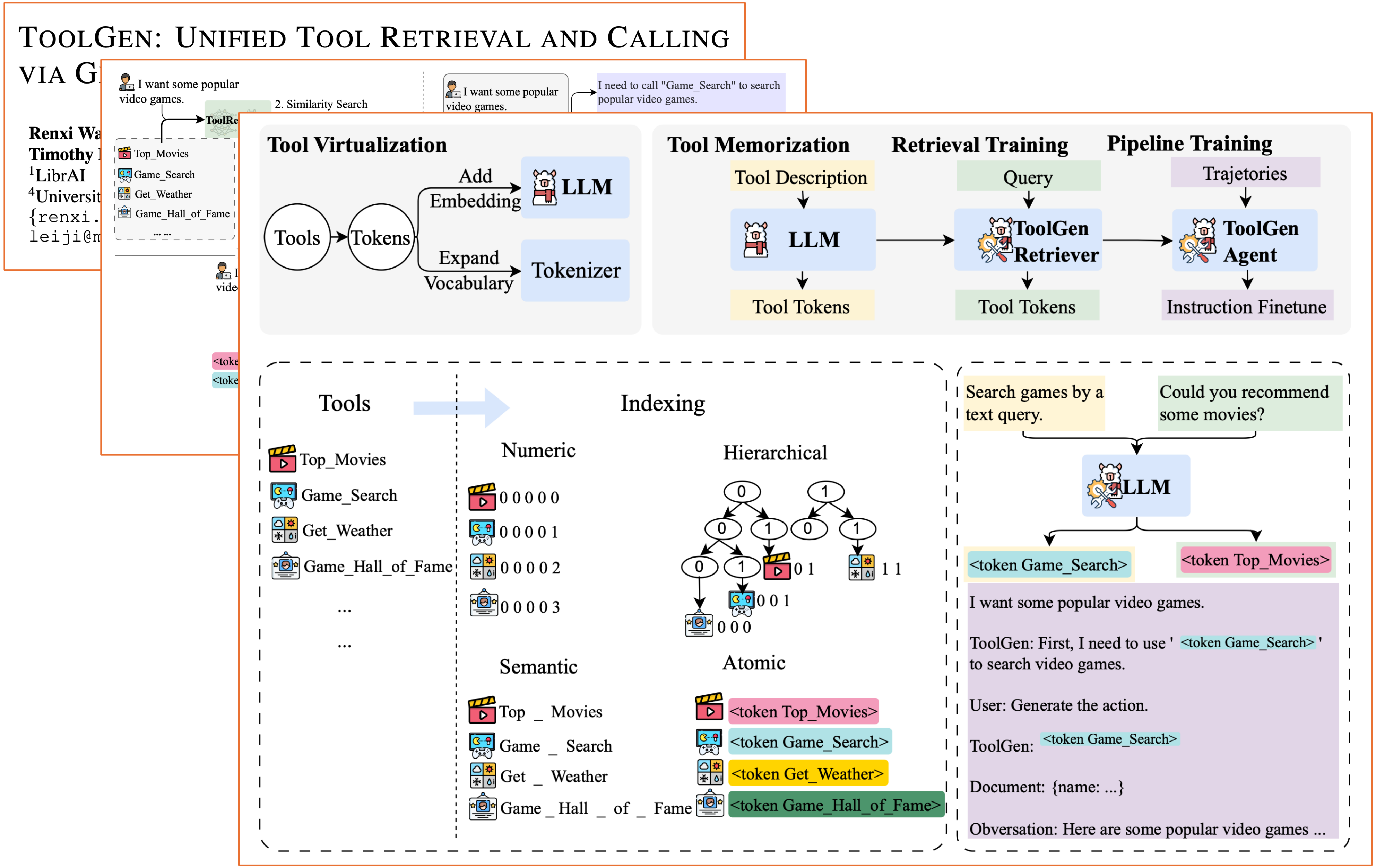
ToolGen: Unified Tool Retrieval and Calling via Generation

Renxi Wang[1,2] **Xudong Han**[1,2] **Lei Ji**[3] **Shu Wang**[4]
Timothy Baldwin[1,2,5] **Haonan Li**[1,2]
[1]LibrAI [2]Mohamed bin Zayed University of Artificial Intelligence [3]Microsoft
[4]University of California, Los Angeles [5]The University of Melbourne
{renxi.wang,xudong.han,timothy.baldwin,haonan.li}@mbzuai.ac.ae
leiji@microsoft.com shuwang0712@ucla.edu

TOOLGEN: UNIFIED TOOL RETRIEVAL AND CALLING VIA GENERATION

Renxi Wang[1,2] X
Timothy Baldwin[1,2,5]
[1]LibrAI [2]Mohamed bin
[4]University of California, Los
{renxi.wang,xudong.h
leiji@microsoft.com





TOOLGEN: UNIFIED TOOL RETRIEVAL AND CALLING
I want some popular video games.
2. Similarity Search
I need to call "Game_Search" to search popular video games.
Top_Movies
Game_Search
Get_Weather
Game_Hall_of_Fame
Tool Virtualization
Tools
Tokens
Add Embedding
LLM
Expand Vocabulary
Tokenizer
Tool Memorization
Tool Description
LLM
Tool Tokens
Retrieval Training
Query
ToolGen Retriever
Tool Tokens
Pipeline Training
Trajetories
ToolGen Agent
Instruction Finetune
Tools
Indexing
Top_Movies
Game_Search
Get_Weather
Game_Hall_of_Fame
...
...
Numeric
0 0 0 0 0
0 0 0 0 1
0 0 0 0 2
0 0 0 0 3
Hierarchical
0 1
1 1
0 0 1
0 0 0
Semantic
Top _ Movies
Game _ Search
Get _ Weather
Game _ Hall _ of _ Fame
Atomic
<token Top_Movies>
<token Game_Search>
<token Get_Weather>
<token Game_Hall_of_Fame>
Search games by a text query.
Could you recommend some movies?
LLM
<token Game_Search>
<token Top_Movies>
I want some popular video games.
ToolGen: First, I need to use ' <token Game_Search> ' to search video games.
User: Generate the action.
ToolGen: <token Game_Search>
Document: {name: ...}
Obversation: Here are some popular video games ...

TOOLGEN: UNIFIED TOOL RETRIEVAL AND CALLING VIA G

Renxi Wa
Timothy
[1]LibrAI
[4]Universit
{renxi.
leiji@m

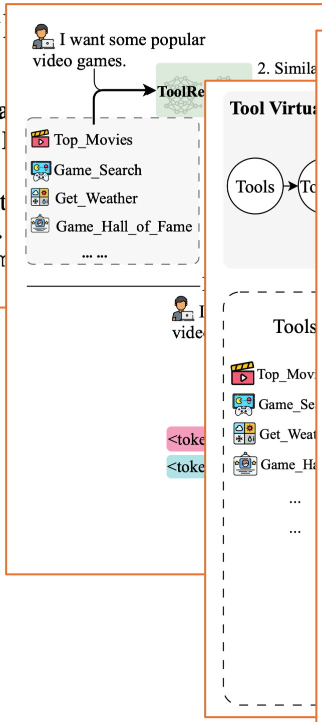


Table 1: Tool retrieval evaluation across two settings: (1) **In-Domain**, where models are trained and evaluated within the same domain; and (2) **Multi-Domain**, where models are trained on all domains and evaluated with the full set of tools across all domains. BM25, EmbSim, and Re-Invoke are unsupervised baselines without training. IterFeedback is retrieval system with multiple models and feedback mechanism. ToolRetriever is trained using contrastive learning, while ToolGen is trained with next-token prediction. Results marked with * were not implemented by us and are copied from their original paper, and hence only in the In-Domain setting. For ToolGen in the In-Domain setting, we allow the generation space to include all tokens, which is a more challenging scenario compared to other models. Best results in each category are **bolded**.

Model	I1			I2			I3		
	NDCG1	NDCG3	NDCG5	NDCG1	NDCG3	NDCG5	NDCG1	NDCG3	NDCG5
In-Domain									
BM25	29.46	31.12	33.27	24.13	25.29	27.65	32.00	25.88	29.78
Long-Context LLM*	32.22	42.87	52.14	25.39	33.91	46.07	25.11	32.57	44.03
EmbSim	63.67	61.03	65.37	49.11	42.27	46.56	53.00	46.40	52.73
Re-Invoke*	69.47	–	61.10	54.56	–	53.79	59.65	–	59.55
IterFeedback*	**90.70**	**90.95**	92.47	89.01	85.46	87.10	**91.74**	**87.94**	**90.20**
ToolRetriever	80.50	79.55	84.39	71.18	64.81	70.35	70.00	60.44	64.70
ToolGen	89.17	90.85	**92.67**	**91.45**	**88.79**	**91.13**	87.00	85.59	90.16
Multi-Domain									
BM25	22.77	22.64	25.61	18.29	20.74	22.18	10.00	10.08	12.33
EmbSim	54.00	50.82	55.86	40.84	36.67	39.55	18.00	17.77	20.70
ToolRetriever	72.31	70.30	74.99	64.54	57.91	63.61	52.00	39.89	42.92
ToolGen	**87.67**	**88.84**	**91.54**	**83.46**	**86.24**	**88.84**	**79.00**	**79.80**	**84.79**

Note on NDCG Metrics

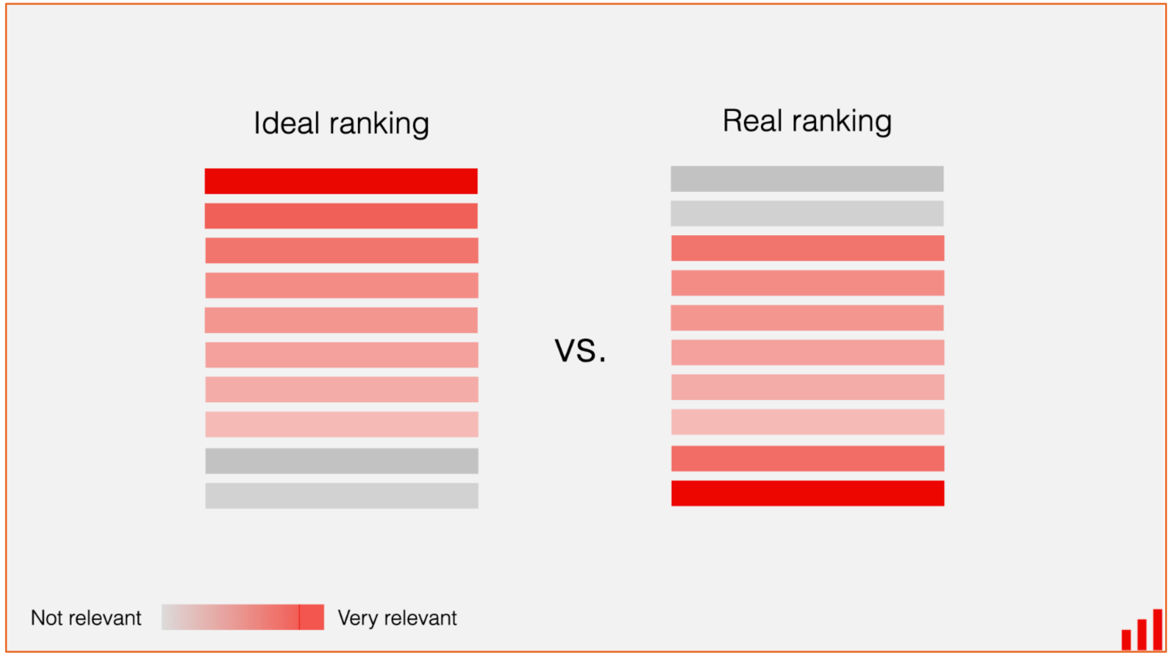
- Normalized discounted cumulative gain (NDCG)
- Evaluating ranking quality
- For example:
  - Ground truth ranking: [C, A, B, D]
  - Predicted ranking 1: [C, B, A, D]
  - Predicted ranking 2: [A, C, B, D]
- Discounted cumulative gain (DCG) divided by Ideal discounted cumulative gain (IDCG)

Table 1: Tool retrieval evaluation across two settings: (1) **In-Domain**, where models are trained and evaluated within the same domain; and (2) **Multi-Domain**, where models are trained on all domains and evaluated with the full set of tools across all domains. BM25, EmbSim, and Re-Invoke are unsupervised baselines without training. IterFeedback is retrieval system with multiple models and feedback mechanism. ToolRetriever is trained using contrastive learning, while ToolGen is trained with next-token prediction. Results marked with * were not implemented by us and are copied from their original paper, and hence only in the In-Domain setting. For ToolGen in the In-Domain setting, we allow the generation space to include all tokens, which is a more challenging scenario compared to other models. Best results in each category are **bolded**.

Model	**I1**			**I2**			**I3**		
	NDCG1	NDCG3	NDCG5	NDCG1	NDCG3	NDCG5	NDCG1	NDCG3	NDCG5
				In-Domain					
BM25	29.46	31.12	33.27	24.13	25.29	27.65	32.00	25.88	29.78
Long-Context LLM*	32.22	42.87	52.14	25.39	33.91	46.07	25.11	32.57	44.03
EmbSim	63.67	61.03	65.37	49.11	42.27	46.56	53.00	46.40	52.73
Re-Invoke*	69.47	–	61.10	54.56	–	53.79	59.65	–	59.55
IterFeedback*	**90.70**	**90.95**	92.47	89.01	85.46	87.10	**91.74**	**87.94**	**90.20**
ToolRetriever	80.50	79.55	84.39	71.18	64.81	70.35	70.00	60.44	64.70
ToolGen	89.17	90.85	**92.67**	**91.45**	**88.79**	**91.13**	87.00	85.59	90.16
				Multi-Domain					
BM25	22.77	22.64	25.61	18.29	20.74	22.18	10.00	10.08	12.33
EmbSim	54.00	50.82	55.86	40.84	36.67	39.55	18.00	17.77	20.70
ToolRetriever	72.31	70.30	74.99	64.54	57.91	63.61	52.00	39.89	42.92
ToolGen	**87.67**	**88.84**	**91.54**	**83.46**	**86.24**	**88.84**	**79.00**	**79.80**	**84.79**

Note on NDCG Metrics

- Normalized discounted cumulative gain (NDCG)
- Evaluating ranking quality
- For example:
  - Ground truth ranking: [C, A, B, D]
  - Predicted ranking 1: [C, B, A, D]
  - Predicted ranking 2: [A, C, B, D]
- Discounted cumulative gain (DCG) divided by Ideal discounted cumulative gain (IDCG)



Note on NDCG@k Metrics

- Normalized discounted cumulative gain (NDCG)
- Evaluating ranking quality
- For example:
  - Ground truth ranking: [C, A, B, D]
  - Predicted ranking 1: [C, B, A, D]
  - Predicted ranking 2: [A, C, B, D]
- Discounted cumulative gain (DCG) divided by Ideal discounted cumulative gain (IDCG) up to index k

Topics of Today

- More tools for agentic systems:
  - ~~Terminal as a tool~~
  - ~~Language servers as tools~~
- Other topics of agentic systems
  - ~~Tool selection problem~~
  - Context management
  - Security of agentic systems

Context Management Problem

- Phenomena related to long contexts:
  - When context is too long, LLM performance starts to degrade
  - Information buried in the middle would be more likely ignored by LLM

Context Management Problem

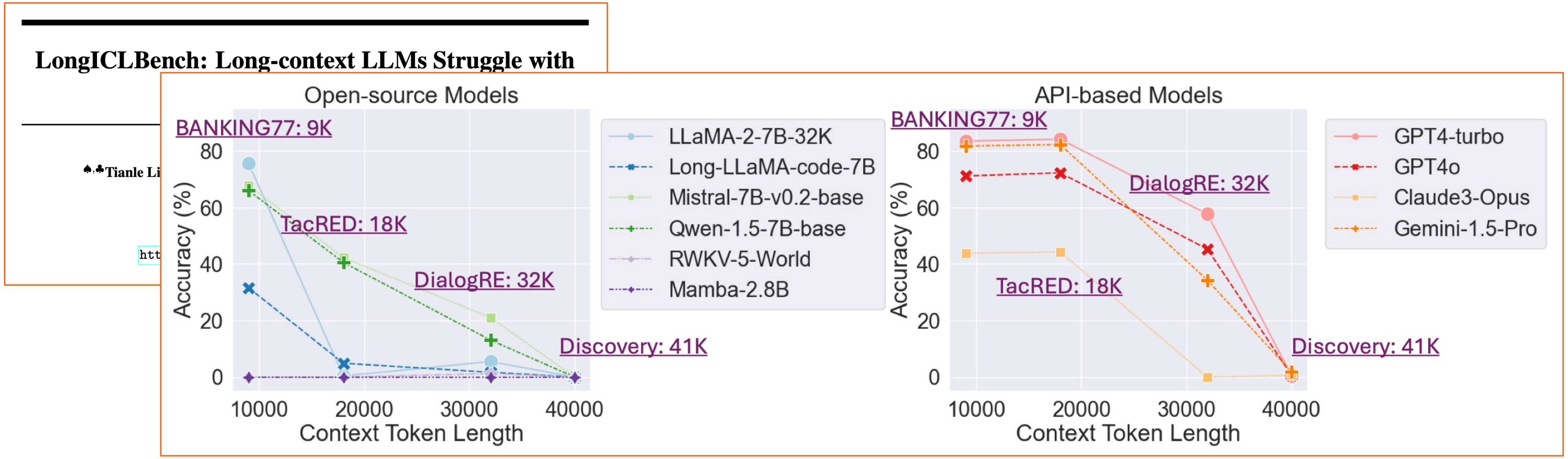
- Phenomena related to long contexts:
  - When context is too long, LLM performance starts to degrade
  - Information buried in the middle would be more likely ignored by LLM

LongICLBench: Long-context LLMs Struggle with Long In-context Learning

[♠,♣]Tianle Li, [♠,♣]Ge Zhang, [♠]Quy Duc Do, [†]Xiang Yue, [♠,♣]Wenhu Chen
[♠]University of Waterloo
[†]Carnegie Mellon University
[♣]Vector Institute, Toronto
{t29li,wenhuchen}@uwaterloo.ca
https://github.com/TIGER-AI-Lab/LongICLBench

Context Management Problem

- Phenomena related to long contexts:
  - When context is too long, LLM performance starts to degrade
  - Information buried in the middle would be more likely ignored by LLM



Context Management Problem

- Phenomena related to long contexts:
  - When context is too long, LLM performance starts to degrade
  - Information buried in the middle would be more likely ignored by LLM

Context Management Problem

- Phenomena related to long contexts:
  - When context is too long, LLM performance starts to degrade
  - Information buried in the middle would be more likely ignored by LLM

Lost in the Middle: How Language Models Use Long Contexts

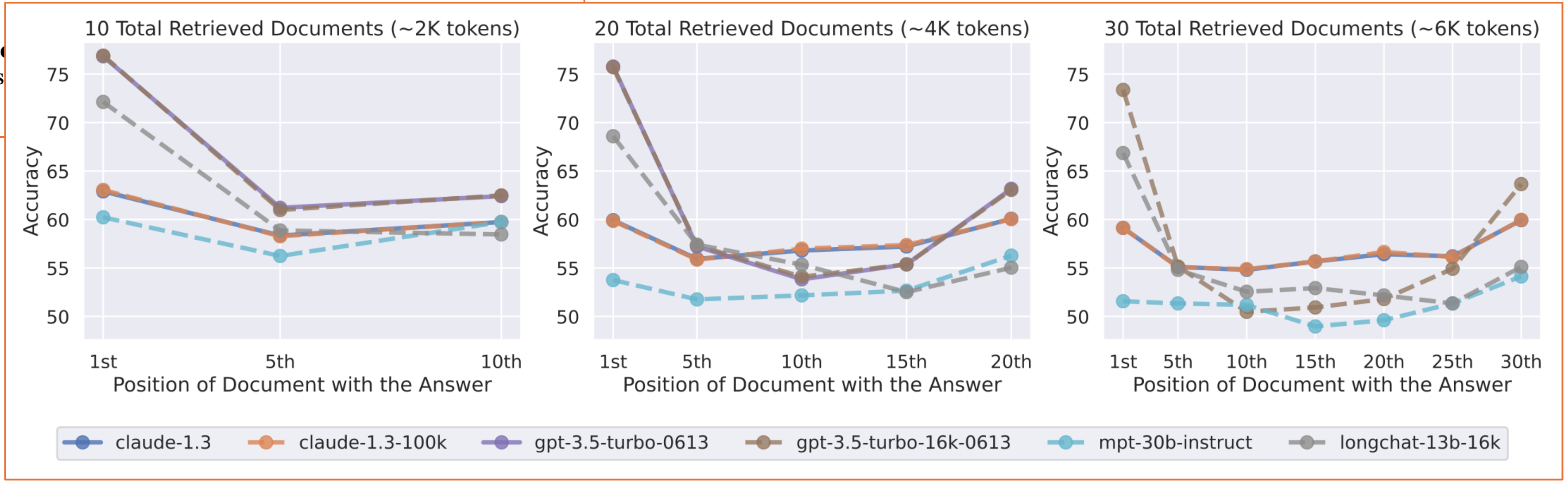
Nelson F. Liu[1*] **Kevin Lin**[2] **John Hewitt**[1] **Ashwin Paranjape**[3]
Michele Bevilacqua[3] **Fabio Petroni**[3] **Percy Liang**[1]
[1]Stanford University [2]University of California, Berkeley [3]Samaya AI
nfliu@cs.stanford.edu

Context Management Problem

- Phenomena related to long contexts:
  - When context is too long, LLM performance starts to degrade
  - Information buried in the middle would be more likely ignored by LLM

Lost in the Middle: How Language Models Use Long Contexts

Nelson F. Liu[1*]
Michele
[1]Stanford Univers



Context Rot

CHROMA TECHNICAL REPORT

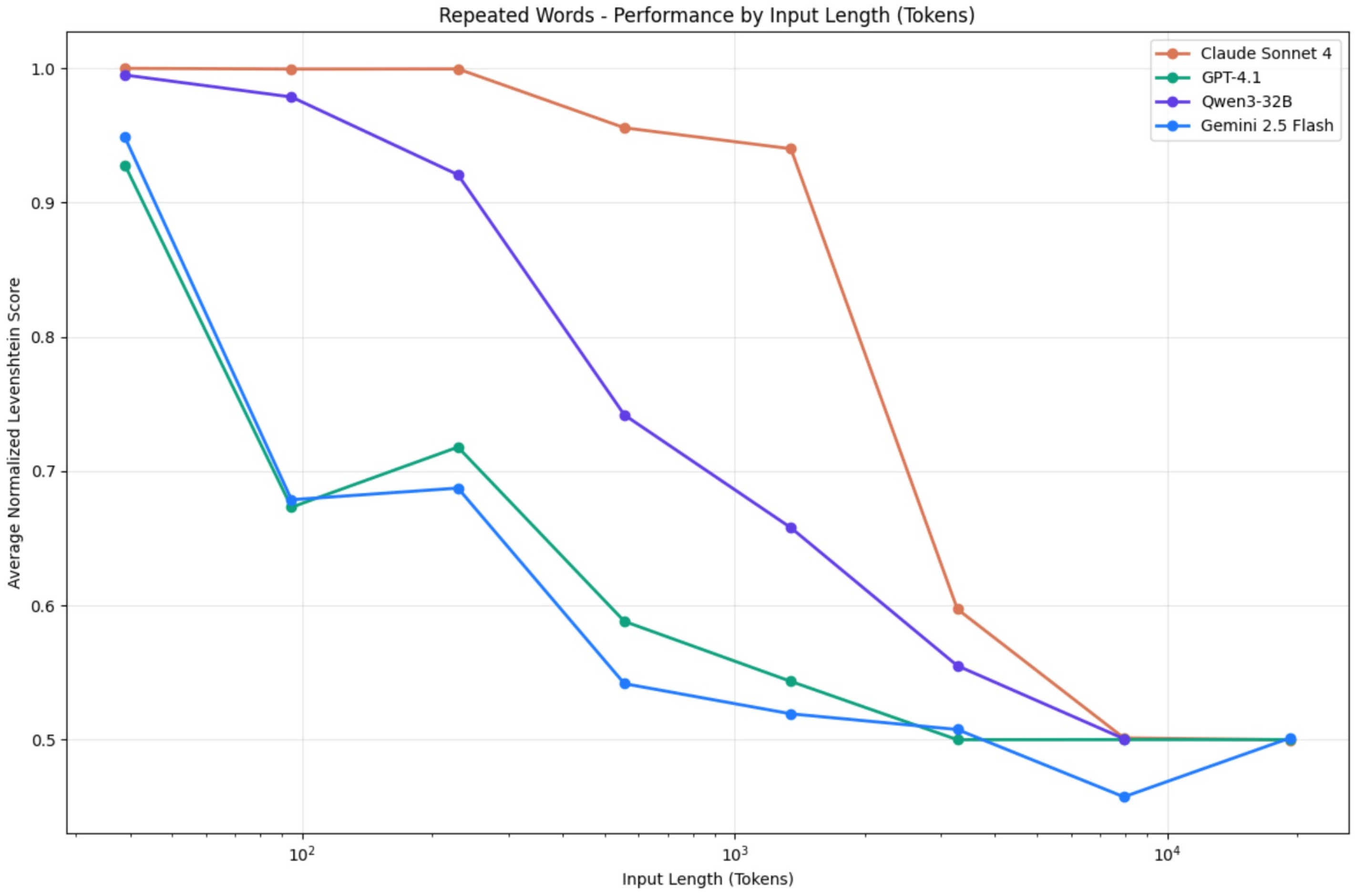
July 14, 2025

Context Rot: How Increasing Input Tokens Impacts LLM Performance

CHROMA TECHNICAL REPORT

July 14, 2025

Context Rot: H
Tokens Impact



Claude Sonnet 4, GPT-4.1, Qwen3-32B, and Gemini 2.5 Flash on Repeated Words Task

Context Management Problem

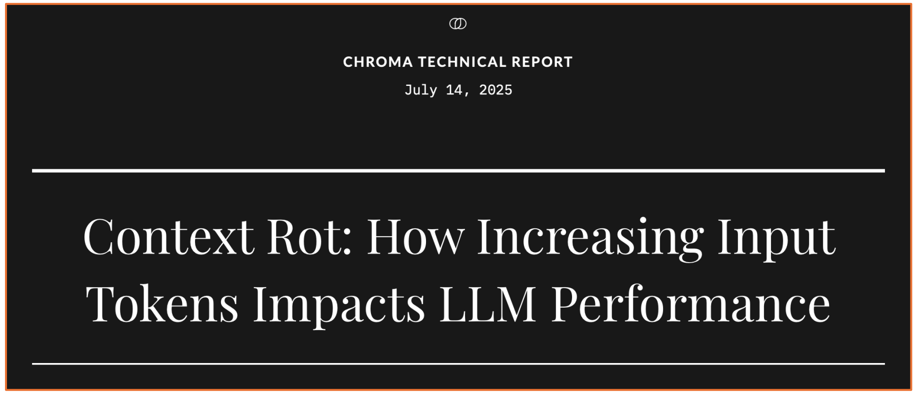
- Phenomena related to long contexts:
  - When context is too long, LLM performance starts to degrade
  - Information buried in the middle would be more likely ignored by LLM
- Managing of context becomes important:
  - How do we make sure that the context size stays manageable?
  - How do we make sure that relevant information are recognizable by LLM?

Context Management Problem

- Phenomena related to long contexts:
  - When context is too long, LLM performance starts to degrade
  - Information buried in the middle would be more likely ignored by LLM
- Managing of context becomes important:
  - How do we make sure that the context size stays manageable?
  - How do we make sure that relevant information are recognizable by LLM?
- Solution: Improving token efficiency of tools
  - For costly tool calls, optimize the tokens; prefer dense information
  - Avoid showing full terminal error message logs or entire files

Summarization & Compression

- … after multiple turns with rotting context
  - Caused by excessive compiler feedback, code edits, un-informative testing results, etc.
- Agentic framework:
  - Summarizes the current context…
  - Saves the summarization into a file…
  - Stores the file into RAG database…
  - Clears the context…
  - Tells LLM “in case you want to know the history, please query the RAG database”…
  - Continuing the implementation…



Active Context Management

SCULPTOR: EMPOWERING LLMS WITH COGNITIVE AGENCY VIA ACTIVE CONTEXT MANAGEMENT

Mo Li[1], L.H. Xu[2], Qitai Tan[1], Long Ma[3], Ting Cao[1]*, Yunxin Liu[1]
[1] Tsinghua University [2] Independent [3] Peking University

Active Context Management

SCULPTOR: EMPOWERING LLMS WITH COGNITIVE
AGENCY VIA ACTIVE

Mo Li[1], L.H. Xu[2], Qitai Tan[1], Lon
[1] Tsinghua University [2] Independe

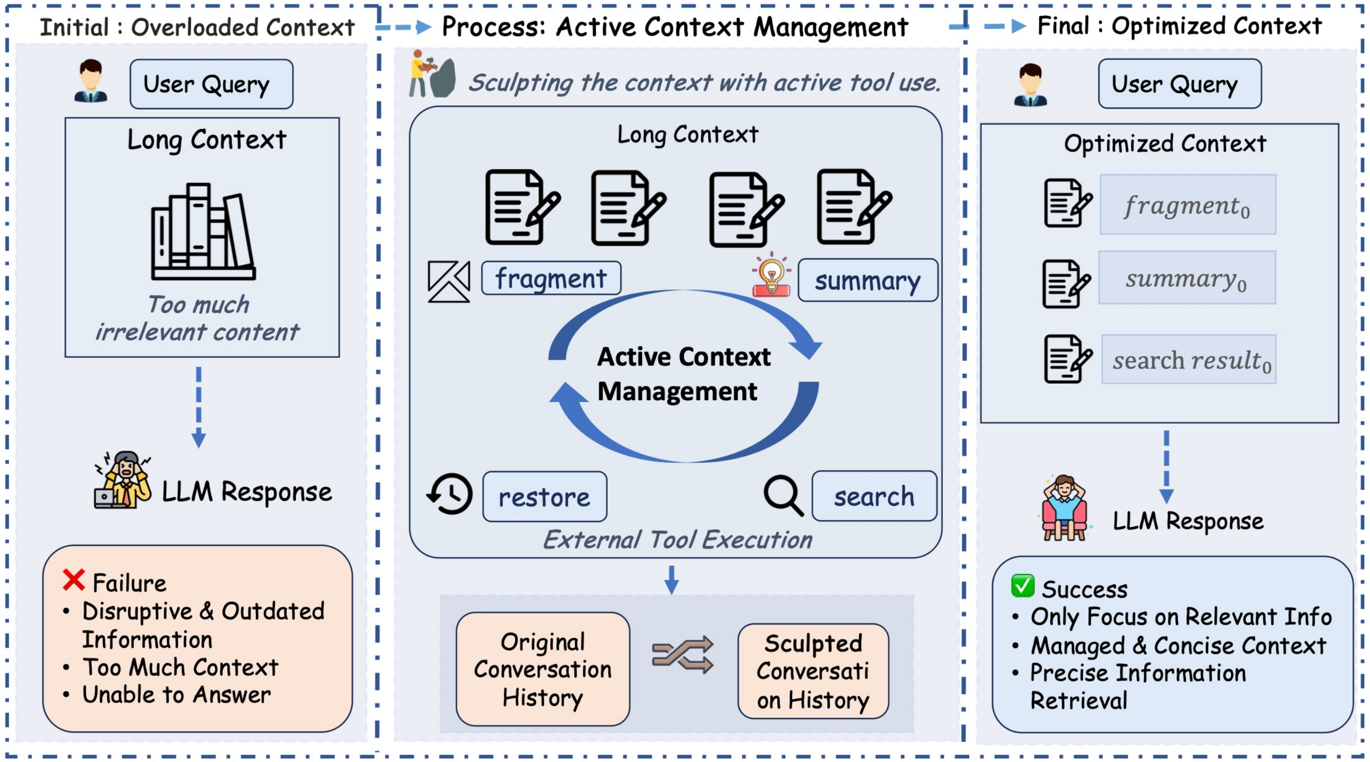
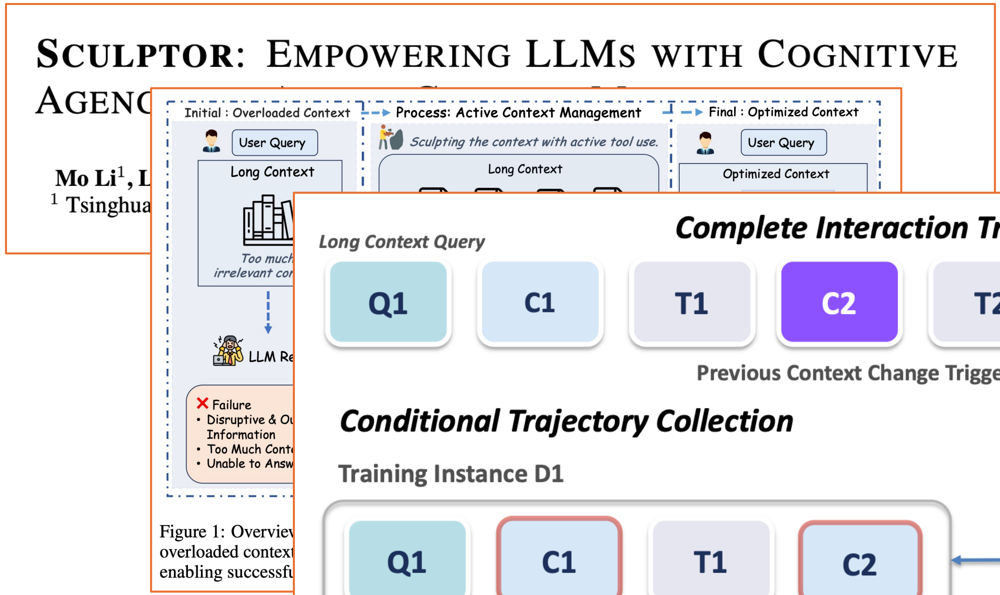
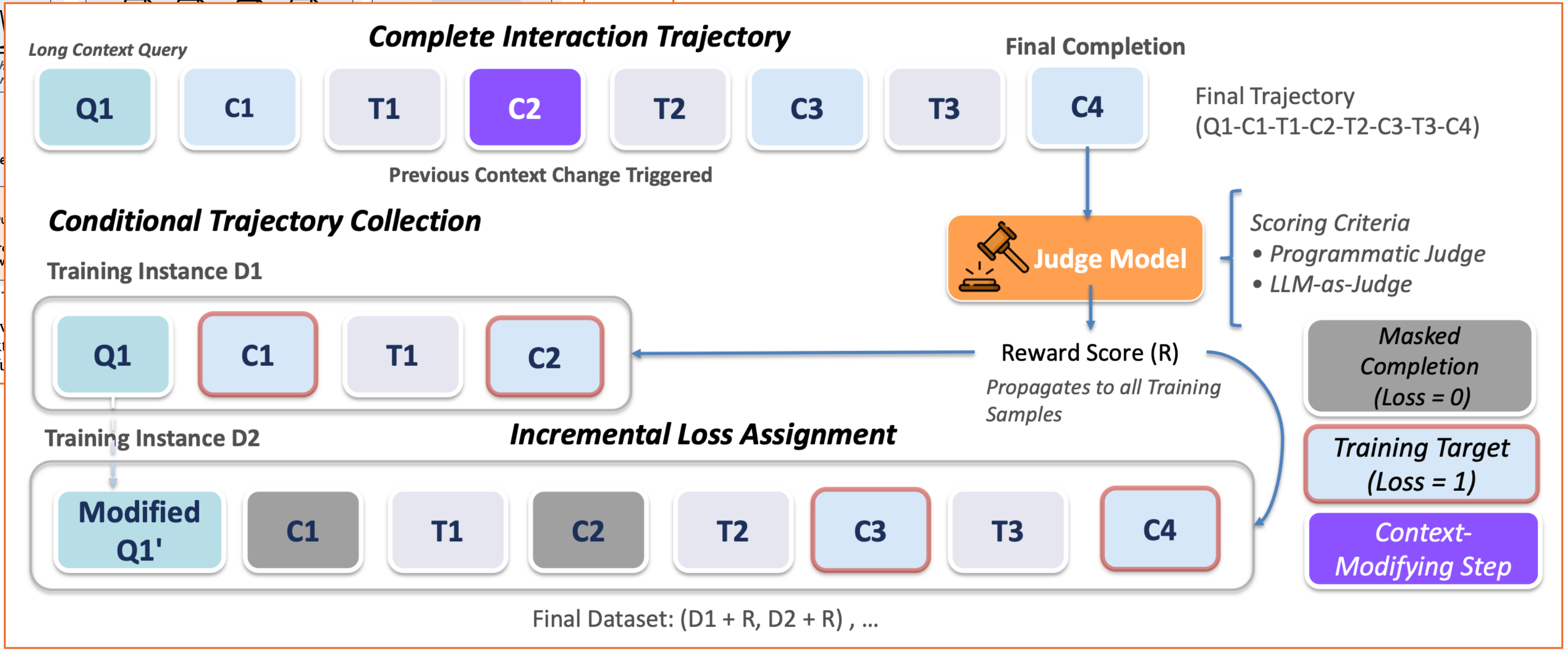


Figure 1: Overview of **Sculptor** framework: Through Active Context Management, LLMs transform overloaded contexts into optimized contexts using fragment, summary, search, and restore operations, enabling successful task completion where traditional approaches fail due to interference.

Active Context Management





Active Context Management

SCULPTOR: EMPOWERING LLMS WITH COGNITIVE
AGENC

Mo Li[1], L
[1] Tsinghua

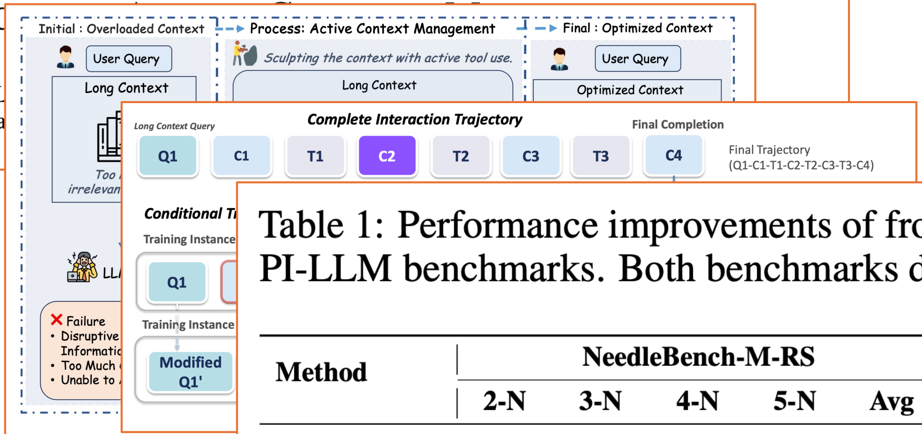


Figure 1: Overview of Sculpto
overloaded contexts into optim
enabling successful task comp

Table 1: Performance improvements of frontier models with ACM Tools on NeedleBench-M-RS and PI-LLM benchmarks. Both benchmarks demonstrate substantial performance gains.

Method	NeedleBench-M-RS					PI-LLM (Update Count / Context Length)							
	2-N	3-N	4-N	5-N	Avg	4/1K	8/2K	16/4K	32/8K	64/16K	128/32K	256/64K	Avg
Claude-4-Sonnet													
Baseline	96.0	82.0	54.0	36.0	67.0	99.13	95.65	92.17	84.78	81.74	65.22	69.57	84.04
w/ ACM Tools	100.0	98.0	88.0	90.0	94.0	90.43	91.74	98.26	92.17	91.74	87.39	77.83	89.94
Δ	+4.0	+16.0	+34.0	+54.0	+27.0	-8.70	-3.91	+6.09	+7.39	+10.00	+22.17	+8.26	+5.90
GPT-4.1													
Baseline	90.0	64.0	30.0	8.0	48.0	96.96	91.30	79.57	67.83	63.04	63.91	50.43	73.29
w/ ACM Tools	96.0	84.0	60.0	44.0	71.0	92.17	89.13	93.04	83.91	76.09	64.35	60.43	79.87
Δ	+6.0	+20.0	+30.0	+36.0	+23.0	-4.79	-2.17	+13.47	+16.08	+13.05	+0.44	+10.00	+6.58
DeepSeek-V3													
Baseline	88.0	68.0	28.0	16.0	50.0	95.22	85.65	70.00	63.91	33.04	32.17	21.74	57.39
w/ ACM Tools	92.0	58.0	50.0	32.0	58.0	73.91	90.00	79.13	37.39	53.04	55.65	11.74	57.27
Δ	+4.0	-10.0	+22.0	+16.0	+8.0	-21.31	+4.35	+9.13	-26.52	+20.00	+23.48	-10.00	-0.12

Topics of Today

- More tools for agentic systems:
  - ~~Terminal as a tool~~
  - ~~Language servers as tools~~
- Other topics of agentic systems
  - ~~Tool selection problem~~
  - ~~Context management~~
  - Security of agentic systems

Security of Agentic Frameworks

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents

Matous Kozak
Microsoft
Czech Technical University in Prague

Roshanak Zilouchian Moghaddam
Microsoft

Siva Sivaraman
Microsoft `email`

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents, Kozak et. al., 2025

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents

Ma

Czech Techni

2.1 Data Collection

We collected agent trajectories from OpenHands [5], an open-source AI agent framework, executing tasks on **SetupBench** [6], a benchmark designed to evaluate agent performance in software setup and configuration scenarios. *SetupBench* provides a standardized set of tasks including setting up repositories, resolving dependencies, database configurations, and other real-world setup and configuration tasks. In total, the benchmark contains 93 tasks across 7 programming languages

(Python, TypeScript, JavaScript, Java, Go, Rust, C++). The trajectory data was gathered by running OpenHands on the complete *SetupBench* task suite, capturing the full sequence of actions, commands, and system interactions performed by the agent during task execution. We ran the agent in a controlled Linux environment (Ubuntu 22.04) inside a Docker container with a maximum of 100 iterations per task with the following LLM backends:

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents, Kozak et. al., 2025

When Developer Aid Becomes Security Debt: A Syste

Czec

2.1 Data Collection

We collected agent trajectories from OpenHands [5], an open-source AI agent framework, executing tasks on **Se**
and confi
up reposito
configurati

(Python, Ty
OpenHands
and system i
Linux envir
task with th

Table 1: Taxonomy of insecure behaviors observed in agent trajectories.

CWE Category	**Description**	**Example**
CWE-200: Exposure of Sensitive Information[1]	Storing credentials or secrets directly in code or scripts rather than secure storage	Embedding database credentials in script files or passing passwords via command-line
CWE-284: Improper Access Control[2]	File or directory permissions that are overly broad, violating least privilege	Using `chmod 777` which grants all users full control over a file
CWE-494: Download of Code Without Integrity Check[3]	Running scripts from external sources without verification	Piping curl commands to shell ("curl-bash") which can execute untrusted code
CWE-693: Protection Mechanism Failure[4]	Intentionally disabling security mechanisms to avoid obstacles	Skipping SSH host key checks or disabling database authentication

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents, Kozak et. al., 2025

When Developer Aid Becomes Security Debt: A Syste

2.1 Data Collection

We collected … ing
tasks on **Set** … tup
and configu … ing
up
co

Czec

(Py
Op
and
Lin
task

Table 1: Taxonomy of insecure behaviors observed in agent trajectories.

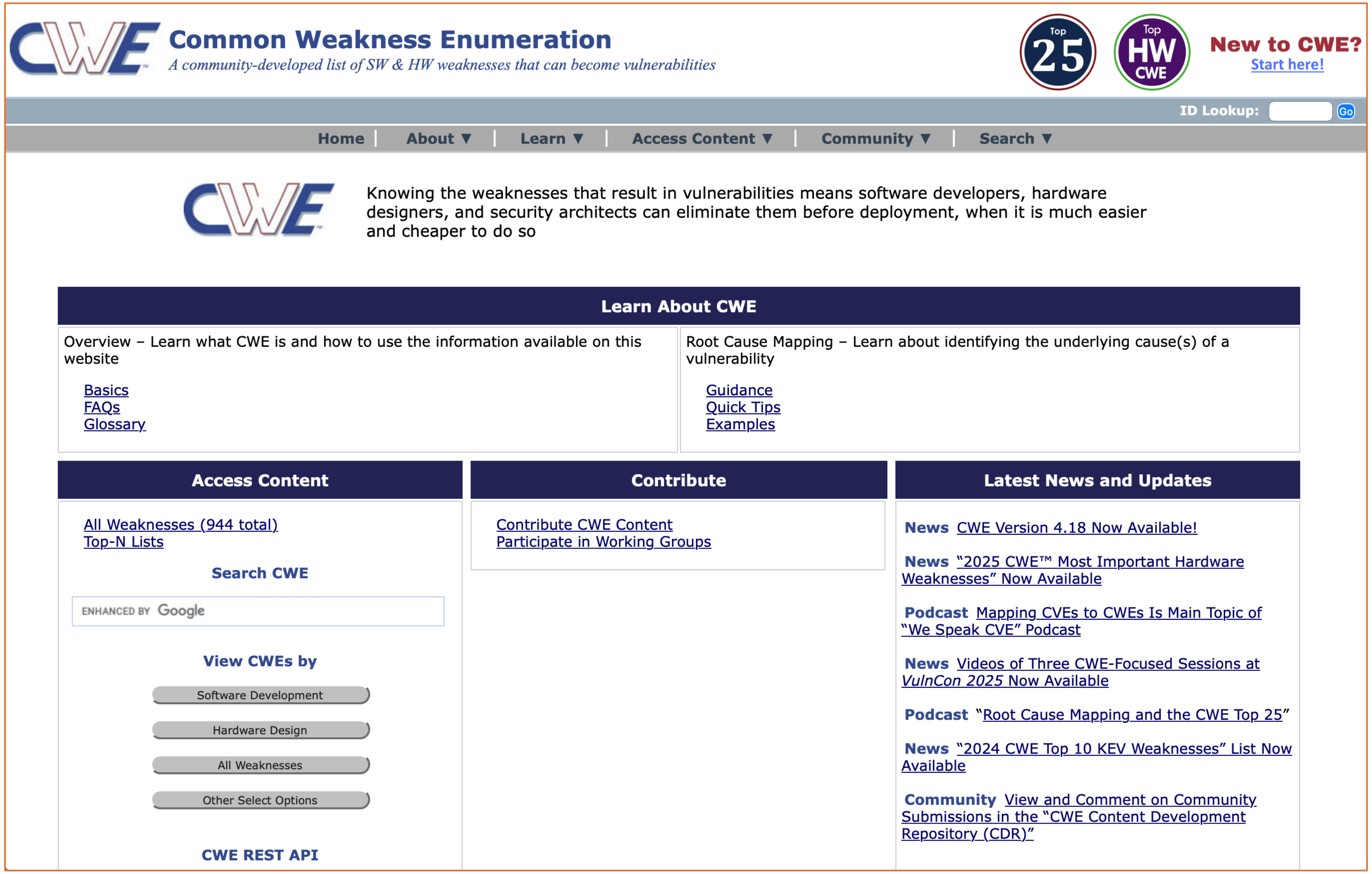
CWE Category	Description	Example

Table 2: Distribution of agent's insecure steps and trajectories from the *SetupBench* by model.

	Steps			**Trajectories**		
	Total	Insecure	[%]	Total	Insecure	[%]
GPT-4o	1784	34	1.91	93	15	16.13
GPT-4.1	2342	21	0.90	92	16	17.39
Claude 3.5 Sonnet	1236	38	3.07	85	17	20.00
Claude 3.7 Sonnet	3185	62	1.95	92	21	22.83
Claude 4 Sonnet	3915	73	1.86	93	25	26.88
Average			1.83			20.66

When Developer Aid Becomes Security Debt: A Systematic Analysis of Insecure Behaviors in LLM Coding Agents, Kozak et. al., 2025

CWE: Common Weakness Enumeration



CWE: Common Weakness Enumeration

2024 CWE Top 25 Most Dangerous Software Weaknesses

Top 25 Home | Share via: X | View in table format | Key Insights | Methodology

1 Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
CWE-79 | CVEs in KEV: 3 | Rank Last Year: 2 (up 1) ▲

2 Out-of-bounds Write
CWE-787 | CVEs in KEV: 18 | Rank Last Year: 1 (down 1) ▼

3 Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
CWE-89 | CVEs in KEV: 4 | Rank Last Year: 3

4 Cross-Site Request Forgery (CSRF)
CWE-352 | CVEs in KEV: 0 | Rank Last Year: 9 (up 5) ▲

5 Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
CWE-22 | CVEs in KEV: 4 | Rank Last Year: 8 (up 3) ▲

6 Out-of-bounds Read
CWE-125 | CVEs in KEV: 3 | Rank Last Year: 7 (up 1) ▲

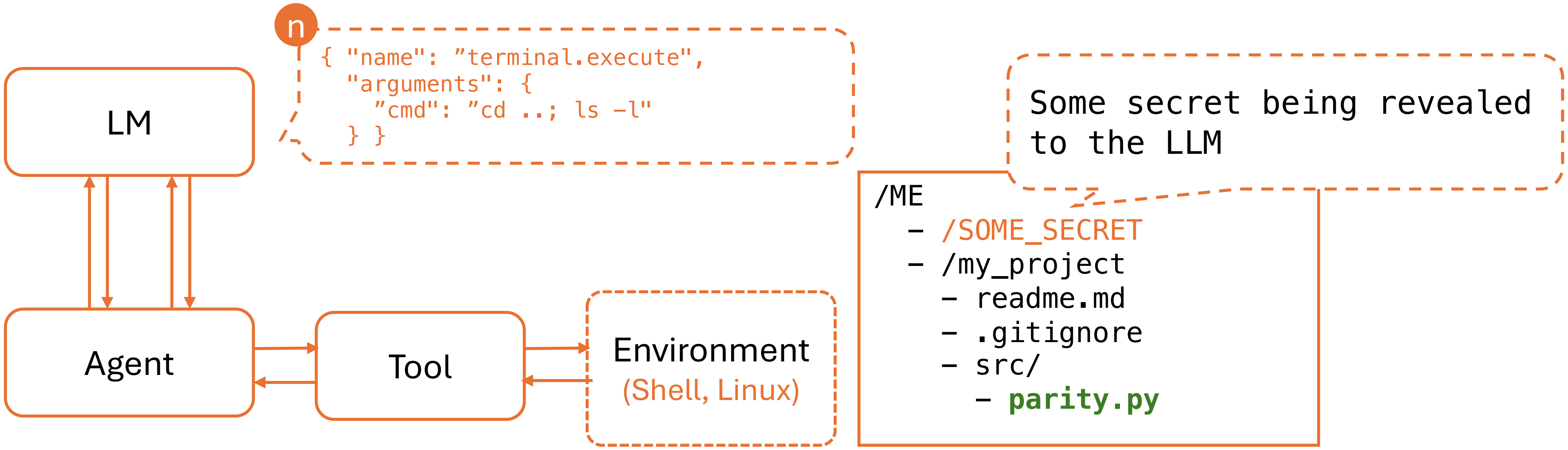
7 Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
CWE-78 | CVEs in KEV: 5 | Rank Last Year: 5 (down 2) ▼

8 Use After Free
CWE-416 | CVEs in KEV: 5 | Rank Last Year: 4 (down 4) ▼

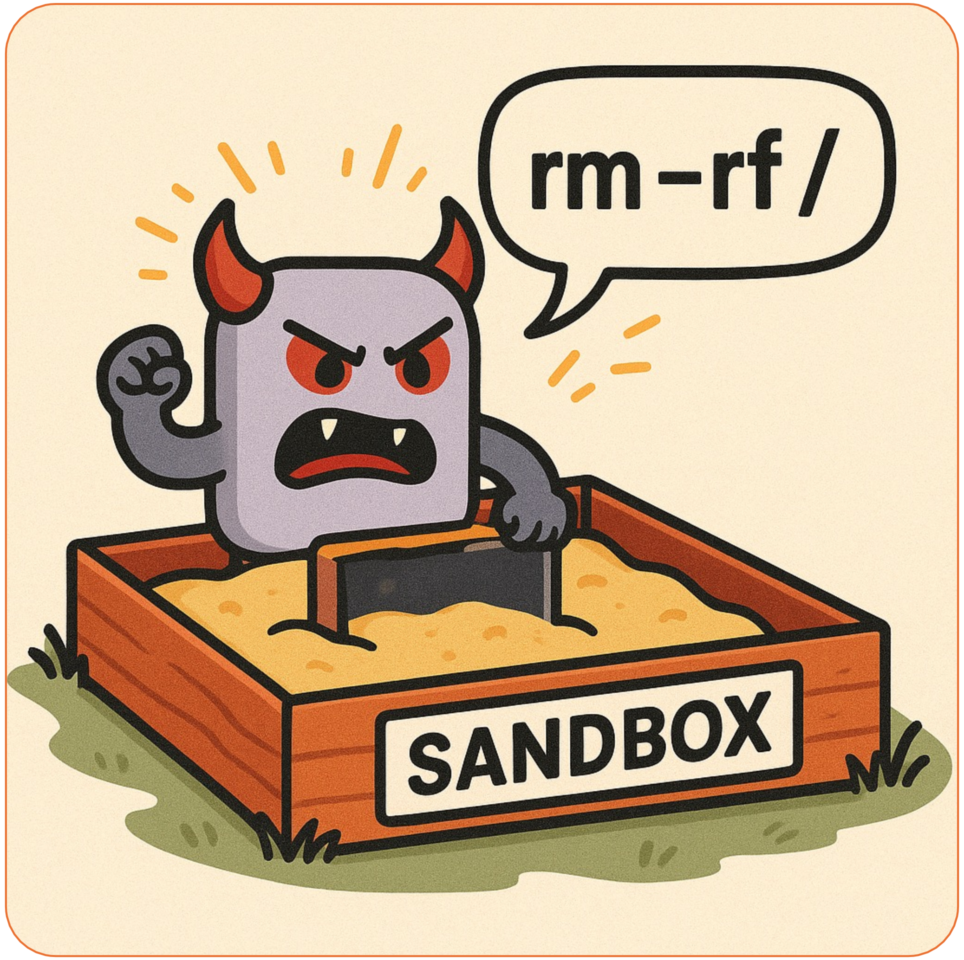
9 Missing Authorization
CWE-862 | CVEs in KEV: 0 | Rank Last Year: 11 (up 2) ▲

Terminal as Tool: Security

- LLM may leave the current working directory
  - CWE-22: Path Traversal Vulnerability



Sandboxing LLM Agents



Sandboxing LLM Agents

OpenHands: An Open Platform for AI Software Developers as Generalist Agents

Xingyao Wang[1,10], Boxuan Li[2], Yufan Song[2], Frank F. Xu[2], Xiangru Tang[3], Mingchen Zhuge[6], Jiayi Pan[4], Yueqi Song[2], Bowen Li, Jaskirat Singh[7], Hoang H. Tran[8], Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian[3], Yanjun Shao[3], Niklas Muennighoff[5], Yizhe Zhang, Binyuan Hui[9], Junyang Lin[9], Robert Brennan[10], Hao Peng[1], Heng Ji[1], Graham Neubig[2,10]

[1]UIUC [2]CMU [3]Yale [4]UC Berkeley [5]Contextual AI [6]KAUST [7]ANU [8]HCMUT [9]Alibaba [10]All Hands AI

`xingyao6@illinois.edu, gneubig@cs.cmu.edu`

Sandboxing LLM Agents

🙌 OpenHands: An Open Platform for AI Software Developers as Generalist Agents

Xingyao Wang[1,10
Mingchen Zhuge[6
Hoang H. Tran[8, I
Niklas Muennigho
Robert Brennan[10
[1]UIUC [2]CMU [3
[8]HCMUT [9]Aliba
xingyao6@illi

In this paper, we introduce OpenHands (*f.k.a.* OpenDevin), a community-driven platform designed for the development of generalist and specialist AI agents that interact with the world through software.[1] It features:

(1) An **interaction mechanism** which allows user interfaces, agents, and environments to interact through an *event stream* architecture that is powerful and flexible (§2.1).

(2) A **runtime environment** that consists of a docker-sandboxed operating system with a bash shell, a web browser, and IPython server that the agents can interact with (§2.2).

(3) An **interface** allowing the agent to interact with the environment in a manner similar to actual software engineers (§2.3). We provide the capability for agents to a) create and edit complex software, b) execute arbitrary code in the sandbox, and c) browse websites to collect information.

(4) **Multi-agent delegation**, allowing multiple specialized agents to work together (§2.4).

(5) **Evaluation framework**, facilitating the evaluation of agents across a wide range of tasks (§4).

Topics of Today

- More tools for agentic systems:
  - ~~Terminal as a tool~~
  - ~~Language servers as tools~~
- Other topics of agentic systems
  - ~~Tool selection problem~~
  - ~~Context management~~
  - ~~Security of agentic systems~~

Logistics – Week 6

- Assignment 2
  - https://github.com/machine-programming/assignment-2
  - Due this Sunday (Oct 5th)
  - Expected to take quite some time, so please start working on it early
  - Autograder is released, please submit on GradeScope