Machine Programming

Lecture 10 – Model Context Protocol for Software Development

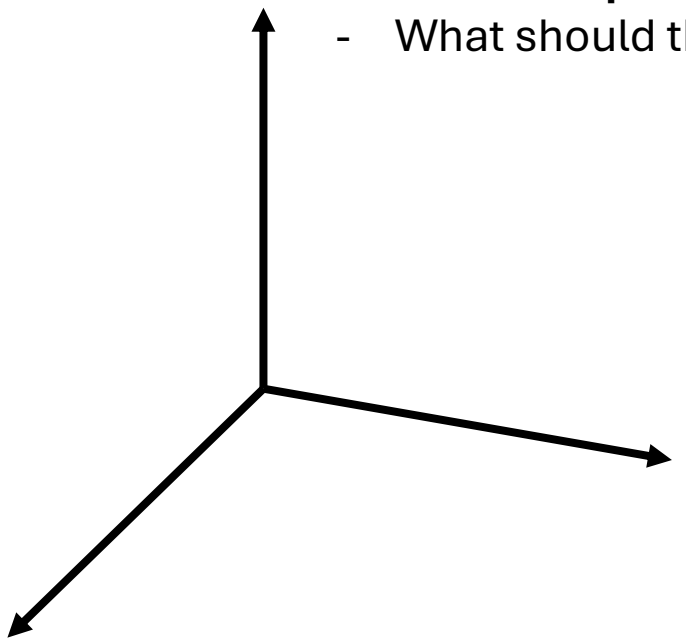
Ziyang Li

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

Dimensions in Program Synthesis

Behavioral Specification
- What should the program do?



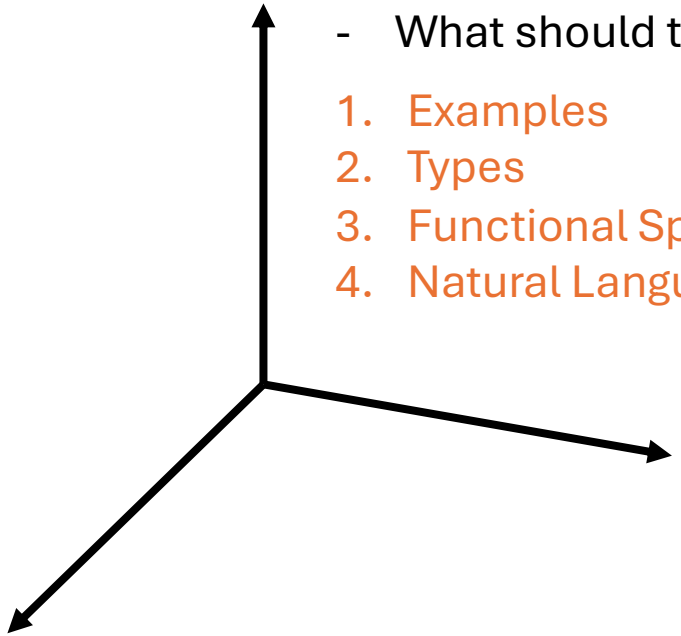
Synthesis Strategy
- How do we find such a program?

Structural Specification
- What is the space of the programs?

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

Structural Specification
- What is the space of the programs?

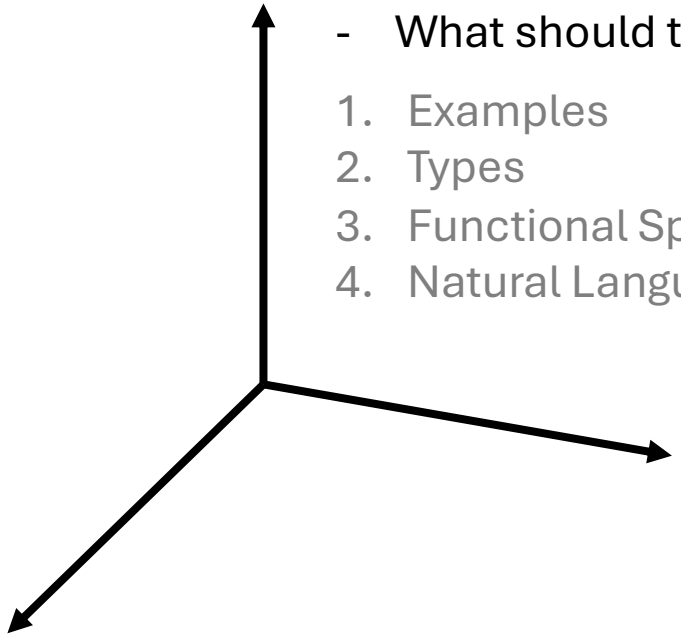
General Purpose Programming Language
Python / Java / C / Rust / …

Domain Specific Languages

Today

Behavioral Specification
- What should the program do?

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



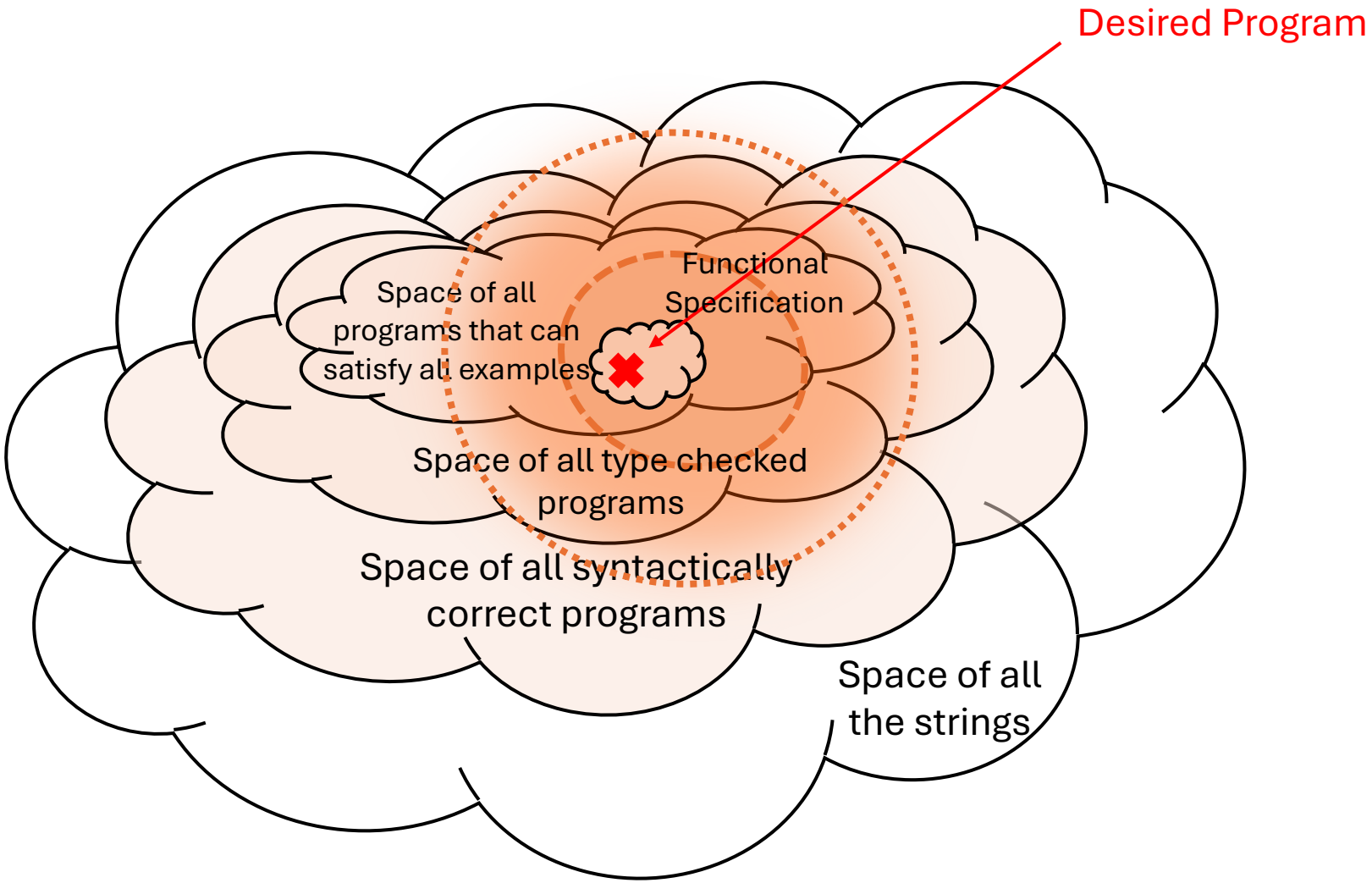
Structural Specification
- What is the space of the programs?

General Purpose Programming Language
Python / Java / C / Rust / …

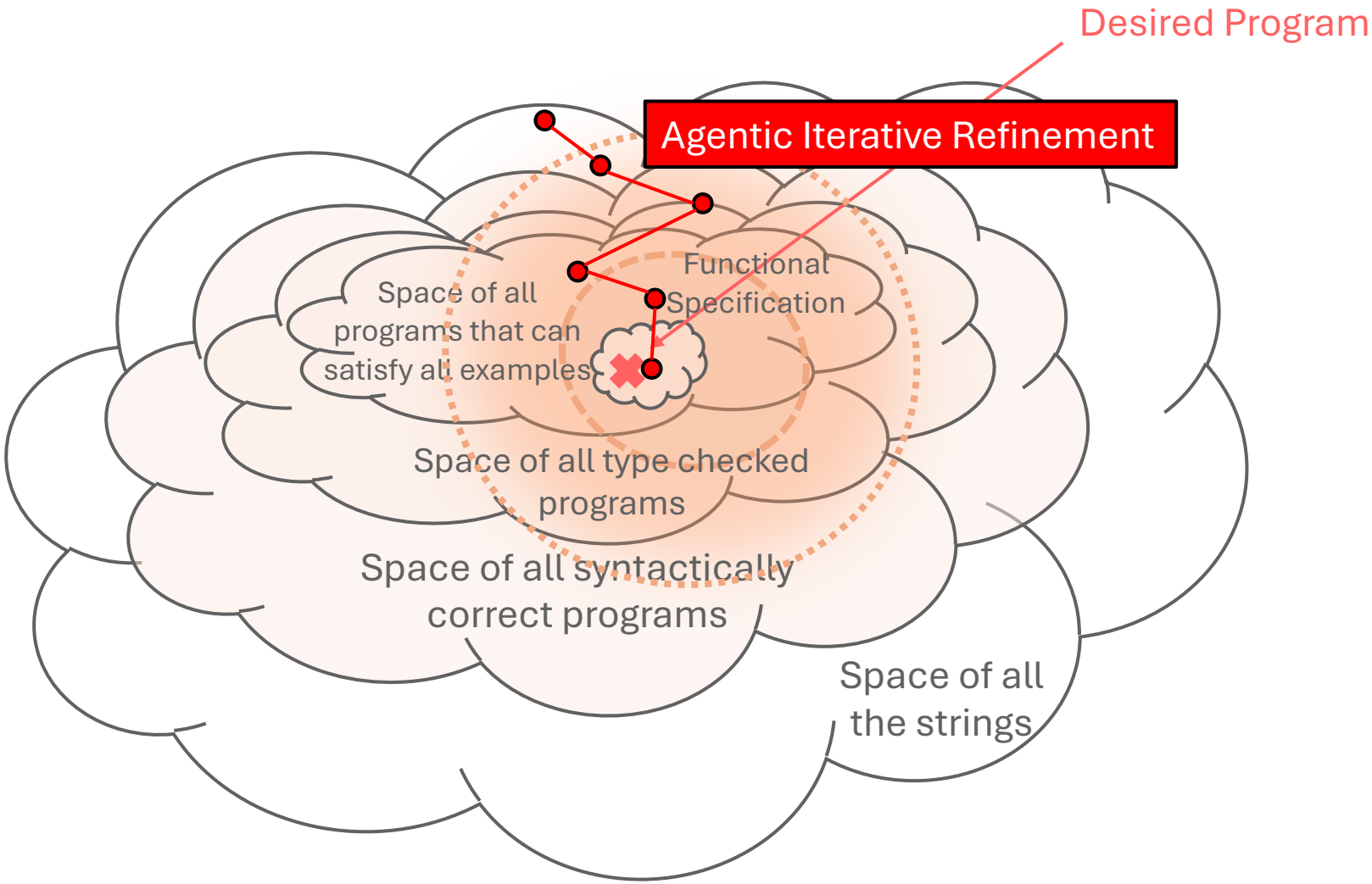
Synthesis Strategy
- How do we find such a program?

Agentic Frameworks
- Tools, MCP Servers, Language Servers

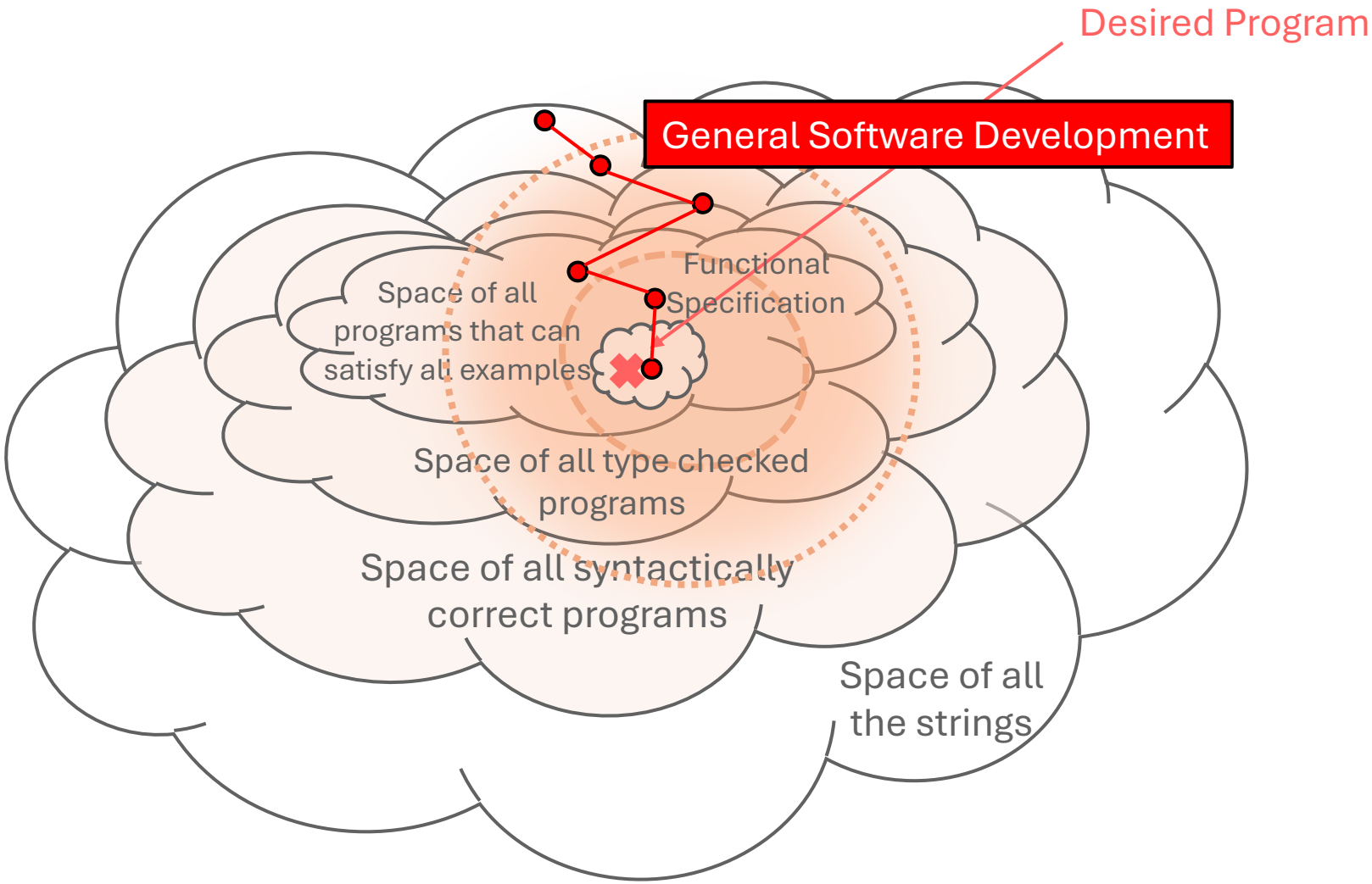
High Level Picture



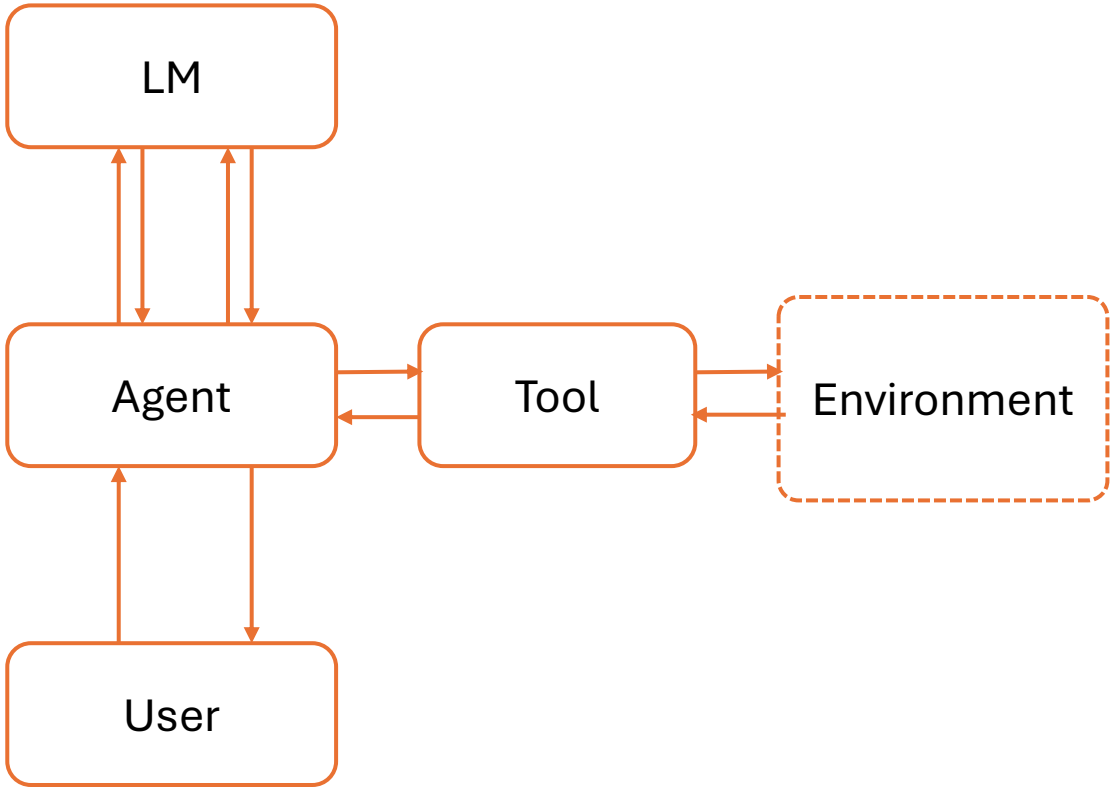
High Level Picture



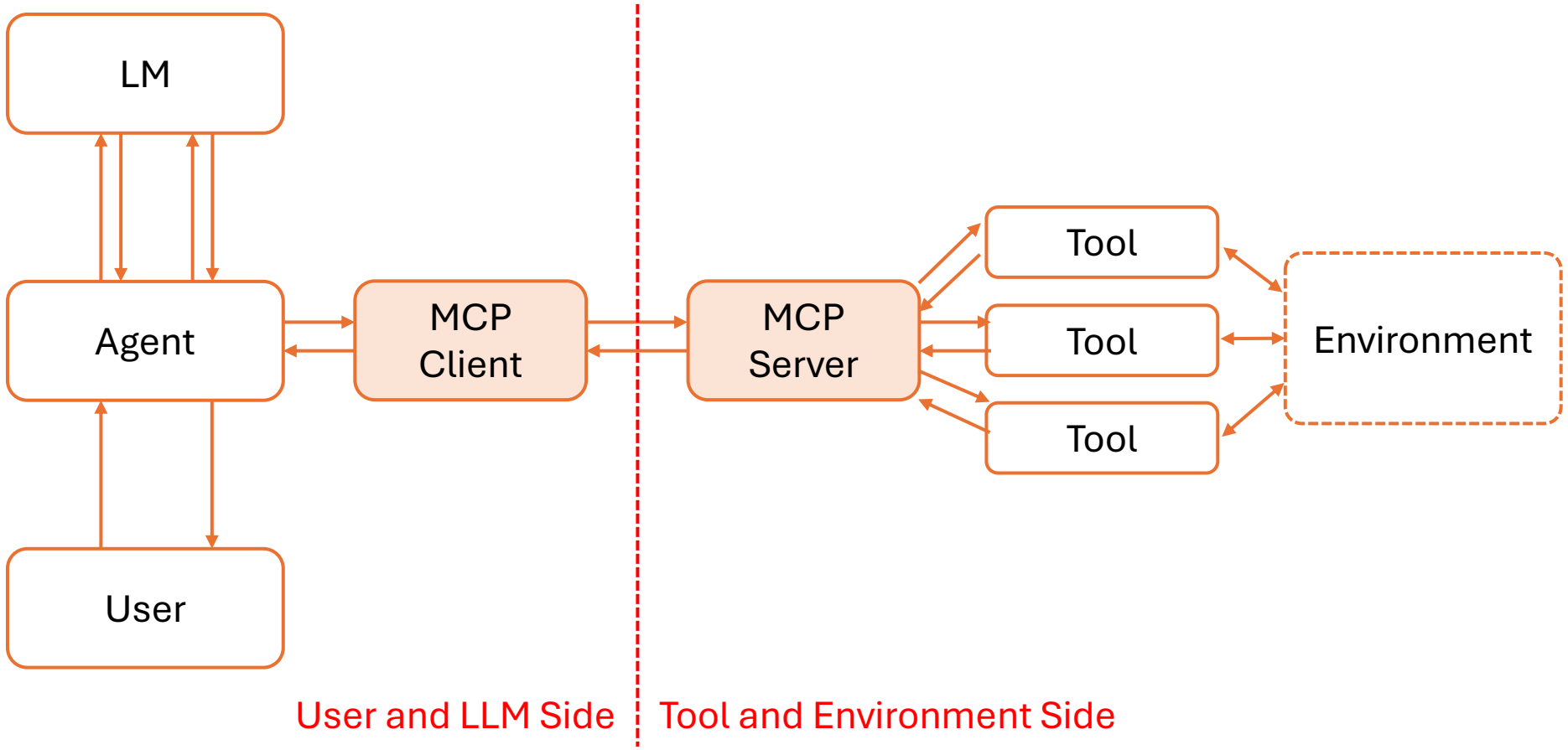
High Level Picture



Agentic Framework



Model Context Protocol (MCP)



Model Context Protocol servers

This repository is a collection of *reference implementations* for the Model Context Protocol (MCP), as well as references to community-built servers and additional resources.

The servers in this repository showcase the versatility and extensibility of MCP, demonstrating how it can be used to give Large Language Models (LLMs) secure, controlled access to tools and data sources. Typically, each MCP server is implemented with an MCP SDK:

https://github.com/modelcontextprotocol/servers

Model Context Protocol servers

This repository is a collection of *reference implementations* for the Model Context Protocol (MCP), as well as references to community-built servers and additional resources.

The servers in this repositor
to give Large Language Mo
server is implemented with

⭐ Reference Servers

These servers aim to demonstrate MCP features and the official SDKs.

- **Everything** - Reference / test server with prompts, resources, and tools.
- **Fetch** - Web content fetching and conversion for efficient LLM usage.
- **Filesystem** - Secure file operations with configurable access controls.
- **Git** - Tools to read, search, and manipulate Git repositories.
- **Memory** - Knowledge graph-based persistent memory system.
- **Sequential Thinking** - Dynamic and reflective problem-solving through thought sequences.
- **Time** - Time and timezone conversion capabilities.

https://github.com/modelcontextprotocol/servers

Model Context Protocol servers

This repo ... reference ... s

The serve ... to give La ... server is i ...

🌟 Reference Servers

These serve ...

- Everyth ...
- Fetch - ...
- Filesys ...
- Git - To ...
- Memor ...
- Sequer ...
- Time - ...

Archived

The following reference servers are now archived and can be found at servers-archived.

- **AWS KB Retrieval** - Retrieval from AWS Knowledge Base using Bedrock Agent Runtime.
- **Brave Search** - Web and local search using Brave's Search API. Has been replaced by the official server.
- **EverArt** - AI image generation using various models.
- **GitHub** - Repository management, file operations, and GitHub API integration.
- **GitLab** - GitLab API, enabling project management.
- **Google Drive** - File access and search capabilities for Google Drive.
- **Google Maps** - Location services, directions, and place details.
- **PostgreSQL** - Read-only database access with schema inspection.
- **Puppeteer** - Browser automation and web scraping.
- **Redis** - Interact with Redis key-value stores.
- **Sentry** - Retrieving and analyzing issues from Sentry.io.
- **Slack** - Channel management and messaging capabilities. Now maintained by Zencoder
- **SQLite** - Database interaction and business intelligence capabilities.

https://github.com/modelcontextprotocol/servers

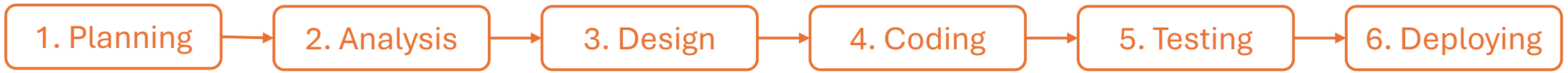
🤝 Third-Party Servers

🎖️ Official Integrations

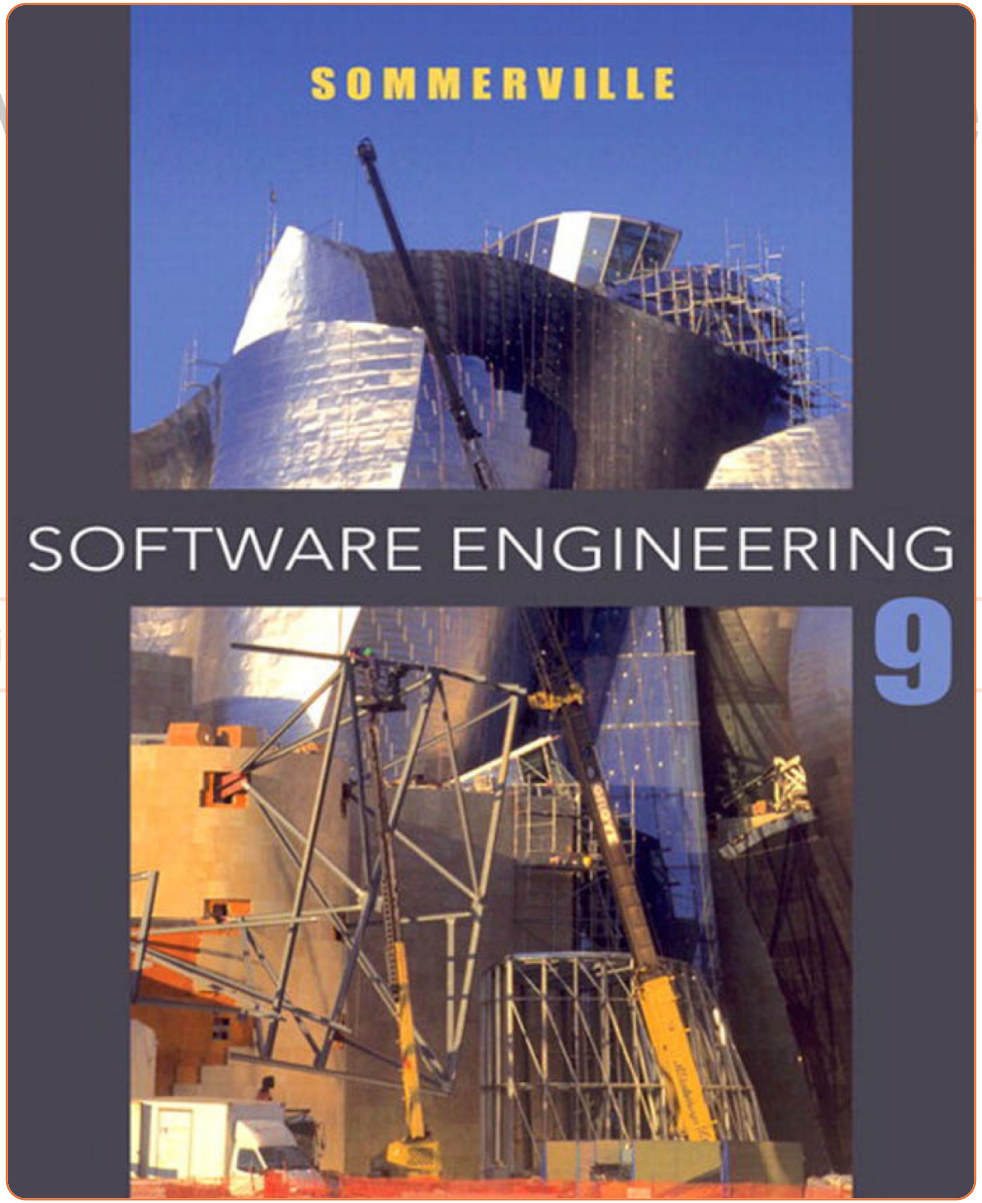
Official integrations are maintained by companies building production ready MCP servers for their platforms.

- **21st.dev Magic** - Create crafted UI components inspired by the best 21st.dev design engineers.
- **ActionKit by Paragon** - Connect to 130+ SaaS integrations (e.g. Slack, Salesforce, Gmail) with Paragon's ActionKit API.
- **Adfin** - The only platform you need to get paid - all payments in one place, invoicing and accounting reconciliations with Adfin.
- **AgentOps** - Provide observability and tracing for debugging AI agents with AgentOps API.
- **AgentQL** - Enable AI agents to get structured data from unstructured web with AgentQL.
- **AgentRPC** - Connect to any function, any language, across network boundaries using AgentRPC.
- **Agentset** - RAG for your knowledge base connected to Agentset.
- **Aiven** - Navigate your Aiven projects and interact with the PostgreSQL®, Apache Kafka®, ClickHouse® and OpenSearch® services
- **Alation** - Unlock the power of the enterprise Data Catalog by harnessing tools provided by the Alation MCP server.
- **Alby Bitcoin Payments** - Connect any bitcoin lightning wallet to your agent to send and receive instant payments globally with your agent.
- **Algolia** - Use AI agents to provision, configure, and query your Algolia search indices.
- **Alibaba Cloud AnalyticDB for MySQL** - Connect to an AnalyticDB for MySQL cluster for getting database or table metadata, querying and analyzing data. It will be supported to add the OpenAPI for cluster operation in the future.
- **Alibaba Cloud AnalyticDB for PostgreSQL** - An MCP server to connect to AnalyticDB for PostgreSQL instances, query and analyze data.
- **Alibaba Cloud DataWorks** - A Model Context Protocol (MCP) server that provides tools for AI, allowing it to interact with the DataWorks Open API through a standardized interface. This implementation is based on the Alibaba Cloud Open API and enables AI agents to perform cloud resources operations seamlessly.
- **Alibaba Cloud OpenSearch** - This MCP server equips AI Agents with tools to interact with OpenSearch through a standardized and extensible interface.
- **Alibaba Cloud OPS** - Manage the lifecycle of your Alibaba Cloud resources with CloudOps Orchestration Service and Alibaba Cloud OpenAPI.
- **Alibaba Cloud RDS** - An MCP server designed to interact with the Alibaba Cloud RDS OpenAPI, enabling programmatic management of RDS resources via an LLM.
- **AlipayPlus** - Connect your AI Agents to AlipayPlus Checkout Payment.
- **AllVoiceLab** - An AI voice toolkit with TTS, voice cloning, and video translation, now available as an MCP server for smarter agent integration.
- **Alpaca** – Alpaca's MCP server lets you trade stocks and options, analyze market data, and build strategies through Alpaca's Trading API
- **AlphaVantage** - Connect to 100+ APIs for financial market data, including stock prices, fundamentals, and more from AlphaVantage
- **AltTester®** - Use AltTester® capabilities to connect and test your Unity or Unreal game. Write game test automation faster and smarter, using AltTester and the AltTester® MCP server.
- **Antom** - Connect your AI Agents to Antom Checkout Payment.
- **Anytype** - An MCP server enabling AI assistants to interact with Anytype - a local and collaborative wiki - to organize objects, lists, and more through natural language.
- **Apache Doris** - MCP Server For Apache Doris, an MPP-based real-time data warehouse.
- **Apache IoTDB** - MCP Server for Apache IoTDB database and its tools
- **Apache Pinot** – MCP server for running real - time analytics queries on Apache Pinot, an open-source OLAP database built for high-throughput, low-latency powering real-time applications.
- **Apify** - Use 6,000+ pre-built cloud tools to extract data from websites, e-commerce, social media, search engines, maps, and more
- **APIMatic MCP** - APIMatic MCP Server is used to validate OpenAPI specifications using APIMatic. The server processes OpenAPI files and returns validation summaries by leveraging APIMatic's API.
- **Apollo MCP Server** - Connect your GraphQL APIs to AI agents
- **Aqara MCP Server** - Control Aqara smart home devices, query status, execute scenes, and much more using natural language.
- **Archbee** - Write and publish documentation that becomes the trusted source for instant answers with AI. Stop cobbling tools and use Archbee — the first complete
- **Astra DB** - Comprehensive tools for managing collections and documents in a DataStax Astra DB NoSQL database with a full range of operations such as create, update, delete, find, and associated bulk actions.
- **Atla** - Enable AI agents to interact with the Atla API for state-of-the-art LLMJ evaluation.
- **Atlan** - The Atlan Model Context Protocol server allows you to interact with the Atlan services through multiple tools.
- **Atlassian** - Securely interact with Jira work items and Confluence pages, and search across both.
- **AtomGit** - Official AtomGit server for integration with repository management, PRs, issues, branches, labels, and more.
- **Audiense Insights** - Marketing insights and audience analysis from Audiense reports, covering demographic, cultural, influencer, and content engagement analysis.
- **Auth0** - MCP server for interacting with your Auth0 tenant, supporting creating and modifying actions, applications, forms, logs, resource servers, and more.
- **Authenticator App · 2FA** - A secure MCP (Model Context Protocol) server that enables AI agents to interact with the Authenticator App.
- **AWS** - Specialized MCP servers that bring AWS best practices directly to your development workflow.
- **Axiom** - Query and analyze your Axiom logs, traces, and all other event data in natural language
- **Azure** - The Azure MCP Server gives MCP Clients access to key Azure services and tools like Azure Storage, Cosmos DB, the Azure CLI, and more.
- **Azure DevOps** - Interact with Azure DevOps services like repositories, work items, builds, releases, test plans, and code search.
- **Backdocket** - Search, Retrieve, and Update your Backdocket data. This currently includes Claims, Matters, Contacts, Tasks and Advanced Searches. To easily use the Remote Mcp Server utilize the following url: **https://ai.backdocket.com/mcp**
- **Baidu Map** - Baidu Map MCP Server provides tools for AI agents to interact with Baidu Maps APIs, enabling location-based services and geospatial data analysis.
- **Bankless Onchain** - Query Onchain data, like ERC20 tokens, transaction history, smart contract state.
- **Baserow** - Query data from Baserow self-hosted or SaaS databases using MCP integration.
- **BICScan** - Risk score / asset holdings of EVM blockchain address (EOA, CA, ENS) and even domain names.
- **Bitrise** - Chat with your builds, CI, and more.
- **Boikot** - Learn about the ethical and unethical actions of major companies with boikot.xyz.
- **BoldSign** - Search, request, and manage e-signature contracts effortlessly with BoldSign.
- **Boost.space** - An MCP server integrating with Boost.space for centralized, automated business data from 2000+ sources.
- **Box** - Interact with the Intelligent Content Management platform through Box AI.
- **BrightData** - Discover, extract, and interact with the web - one interface powering automated access across the public internet.
- **Browserbase** - Automate browser interactions in the cloud (e.g. web navigation, data extraction, form filling, and more)
- **BrowserStack** - Access BrowserStack's Test Platform to debug, write and fix tests, do accessibility testing and more.
- **Buildkite** - Exposing Buildkite data (pipelines, builds, jobs, tests) to AI tooling and editors.
- **Buildable** (TypeScript) - Official MCP server for Buildable AI-powered development platform. Enables AI assistants to manage tasks, track progress, get project context, and collaborate with humans on software projects.
- **BuiltWith** - Identify the technology stack behind any website.
- **Burp Suite** - MCP Server extension allowing AI clients to connect to Burp Suite
- **Cal.com** - Connect to the Cal.com API to schedule and manage bookings and appointments.
- **Campertunity** - Search campgrounds around the world on campertunity, check availability, and provide booking links.
- **Canva** — Provide AI - powered development assistance for Canva apps and integrations.
- **Carbon Voice** - MCP Server that connects AI Agents to Carbon Voice. Create, manage, and interact with voice messages, conversations, direct messages, folders, voice memos, AI actions and more in Carbon Voice.
- **Cartesia** - Connect to the Cartesia voice platform to perform text-to-speech, voice cloning etc.
- **Cashfree** - Cashfree Payments official MCP server.
- **CB Insights** - Use the CB Insights MCP Server to connect to ChatCBI
- **Cloudinary** - Exposes Cloudinary's media upload, transformation, AI analysis, management, optimization and delivery as tools usable by AI agents
- **Cloudsway SmartSearch** - Web search MCP server powered by Cloudsway, supporting keyword search, language, and safety options. Returns structured JSON results.
- **Codacy** - Interact with Codacy API to query code quality issues, vulnerabilities, and coverage insights about your code.
- **CodeLogic** - Interact with CodeLogic, a Software Intelligence platform that graphs complex code and data architecture dependencies, to boost AI accuracy and insight.
- **CoinGecko** - Official CoinGecko API MCP Server for Crypto Price & Market Data, across 200+ Blockchain Networks and 8M+ Tokens.
- **Comet Opik** - Query and analyze your Opik logs, traces, prompts and all other telemetry data from your LLMs in natural language.
- **Conductor** - Interact with Conductor (OSS and Orkes) REST APIs.
- **Composio** – Use Composio to connect 100+ tools. Zero setup. Auth built-in. Made for agents, works for humans.
- **Confluent** - Interact with Confluent Kafka and Confluent Cloud REST APIs.
- **Contrast Security** - Brings Contrast's vulnerability and SCA data into your coding agent to quickly remediate vulnerabilities.
- **Convex** - Introspect and query your apps deployed to Convex.
- **Cortex** - Official MCP server for Cortex.
- **Couchbase** - Interact with the data stored in Couchbase clusters.
- **CRIC Wuye AI** - Interact with capabilities of the CRIC Wuye AI platform, an intelligent assistant specifically for the property management industry.
- **CrowdStrike Falcon** - Connects AI agents with the CrowdStrike Falcon platform for intelligent security analysis, providing programmatic access to detections, incidents, behaviors, threat intelligence, hosts, vulnerabilities, and identity protection capabilities.
- **CTERA Edge Filer** - CTERA Edge Filer delivers intelligent edge caching and multiprotocol file access, enabling fast, secure access to files across core and remote sites.
- **CTERA Portal** - CTERA Portal is a multi-tenant, multi-cloud platform that delivers a global namespace and unified management across petabytes of distributed content.
- **Cycode** - Boost security in your dev lifecycle via SAST, SCA, Secrets & IaC scanning with Cycode.
- **Dart** - Interact with task, doc, and project data in Dart, an AI-native project management tool
- **Databricks** - Connect to data, AI tools & agents, and the rest of the Databricks platform using turnkey managed MCP servers. Or, host your own custom MCP servers within the Databricks security and data governance boundary.
- **DataHub** - Search your data assets, traverse data lineage, write SQL queries, and more using DataHub metadata.
- **Daytona** - Fast and secure execution of your AI generated code with Daytona sandboxes
- **Debugg.AI** - Zero-Config, Fully AI-Managed End-to-End Testing for any code gen platform via Debugg.AI remote browsing test agents.
- **DeepL** - Translate or rewrite text with DeepL's very own AI models using the DeepL API
- **Defang** - Deploy your project to the cloud seamlessly with the Defang platform without leaving your integrated development environment
- **Detailer** – Instantly generate rich, AI-powered documentation for your GitHub repositories. Designed for AI agents to gain deep project context before taking action.
- **DevCycle** - Create and monitor feature flags using natural language in your AI coding assistant.
- **DevHub** - Manage and utilize website content within the DevHub CMS platform
- **DevRev** - An MCP server to integrate with DevRev APIs to search through your DevRev Knowledge Graph where objects can be imported from diff. Sources listed here.
- **DexPaprika (CoinPaprika)** - Access real-time DEX data, liquidity pools, token information, and trading analytics across multiple blockchain networks with DexPaprika by CoinPaprika.
- **Dolt** - The official MCP server for version-controlled Dolt databases.
- **Dot (GetDot.ai)** - Fetch, analyze or visualize data from your favorite database or data warehouse (Snowflake, BigQuery, Redshift, Databricks, Clickhouse, ...) with Dot, your AI Data Analyst. This remote MCP server is a one-click integration for user that have setup Dot.
- **Drata** - Get hands-on with our experimental MCP server—bringing real-time compliance intelligence into your AI workflows.
- **Dumpling AI** - Access data, web scraping, and document conversion APIs by Dumpling AI
- **GitGuardian** - GitGuardian official MCP server - Scan projects using GitGuardian's industry-leading API, which features over 500 secret detectors to prevent credential leaks before they reach public repositories. Resolve security incidents directly with rich contextual data for rapid, automated remediation.
- **GitHub** - GitHub's official MCP Server.
- **GitKraken** - A CLI for interacting with GitKraken APIs. Includes an MCP server via `gk mcp` that not only wraps GitKraken APIs, but also Jira, GitHub, GitLab, and more.
- **Glean** - Enterprise search and chat using Glean's API.
- **Globalping** - Access a network of thousands of probes to run network commands like ping, traceroute, mtr, http and DNS resolve.
- **gNucleus Text-To-CAD** - Generate CAD parts and assemblies from text using gNucleus AI models.
- **Google Cloud Run** - Deploy code to Google Cloud Run
- **GoLogin MCP server** - Manage your GoLogin browser profiles and automation directly through AI conversations!
- **Google Maps Platform Code Assist** - Ground agents on fresh, official documentation and code samples for optimal geo-related guidance and code..
- **gotoHuman** - Human-in-the-loop platform - Allow AI agents and automations to send requests for approval to your gotoHuman inbox.
- **Grafana** - Search dashboards, investigate incidents and query datasources in your Grafana instance
- **Grafbase** - Turn your GraphQL API into an efficient MCP server with schema intelligence in a single command.
- **Grain** - Access your Grain meetings notes & transcripts directly in claude and generate reports with native Claude Prompts.
- **Graphlit** - Ingest anything from Slack to Gmail to podcast feeds, in addition to web crawling, into a searchable Graphlit project.
- **Gremlin** - The official Gremlin MCP server. Analyze your reliability posture, review recent tests and chaos engineering experiments, and create detailed reports.
- **GreptimeDB** - Provides AI assistants with a secure and structured way to explore and analyze data in GreptimeDB.
- **GROWI** - Official MCP Server to integrate with GROWI APIs.
- **Gyazo** - Search, fetch, upload, and interact with Gyazo images, including metadata and OCR data.
- **Harper** - An MCP server providing an interface for MCP clients to access data within Harper.
- **Heroku** - Interact with the Heroku Platform through LLM-driven tools for managing apps, add-ons, dynos, databases, and more.
- **HeyOnCall** - Page a human, sending critical or non-critical alerts to the free HeyOnCall iOS or Android apps.
- **Hiveflow** - Create, manage, and execute agentic AI workflows directly from your assistant.
- **Hive Intelligence** - Ultimate cryptocurrency MCP for AI assistants with unified access to crypto, DeFi, and Web3 analytics
- **Hologres** - Connect to a Hologres instance, get table metadata, query and analyze data.
- **Homebrew** Allows Homebrew users to run Homebrew commands locally.
- **Honeycomb** Allows Honeycomb Enterprise customers to query and analyze their data, alerts, dashboards, and more; and cross-reference production behavior with the codebase.
- **HubSpot** - Connect, manage, and interact with HubSpot CRM data
- **Hugging Face** - Connect to the Hugging Face Hub APIs programmatically: semantic search for spaces and papers, exploration of datasets and models, and access to all compatible MCP Gradio tool spaces!
- **Hunter** - Interact with the Hunter API to get B2B data using natural language.
- **Hyperbolic** - Interact with Hyperbolic's GPU cloud, enabling agents and LLMs to view and rent available GPUs, SSH into them, and run GPU-powered workloads for you.
- **Hyperbrowser** - Hyperbrowser is the next-generation platform empowering AI agents and enabling effortless, scalable browser automation.
- **IBM wxflows** - Tool platform by IBM to build, test and deploy tools for any data source
- **Inbox Zero** - AI personal assistant for email Inbox Zero
- **Inflectra Spira** - Connect to your instance of the SpiraTest, SpiraTeam or SpiraPlan application lifecycle management platform by Inflectra
- **Inkeep** - RAG Search over your content powered by Inkeep
- **Integration App** - Interact with any other SaaS applications on behalf of your customers.
- **IP2Location.io** - Interact with IP2Location.io API to retrieve the geolocation information for an IP address.
- **IPLocate** - Look up IP address geolocation, network information, detect proxies and VPNs, and find abuse contact details using IPLocate.io

Software Development Life Cycle (SDLC)

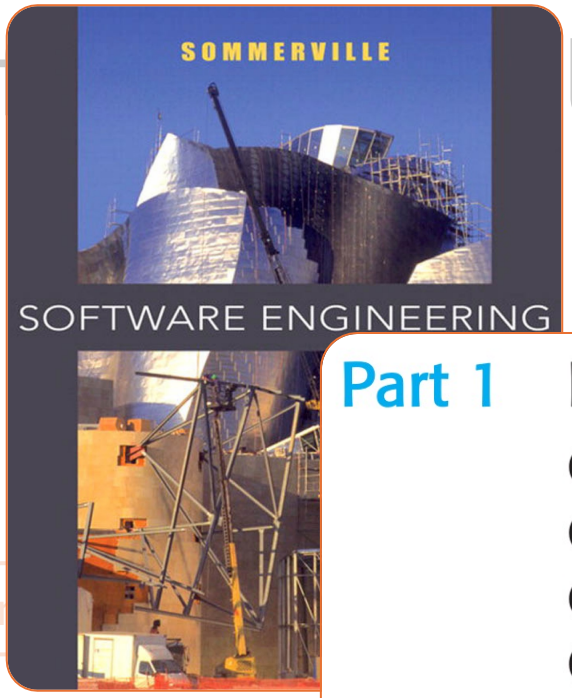


Software Dev (SDLC)



1. Planning → 2. Analysi → 5. Testing → 6. Deploying

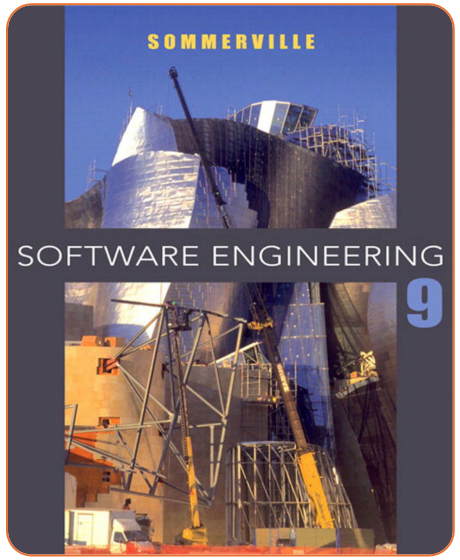
Software Development Life Cycle (SDLC)



1. Pla

6. Deploying

Part 1 Introduction to Software Engineering



Development Life Cycle (SDLC)

1. Planning → 2. Analysis → 3. Design → 4. Coding → 5. Testing → 6. Deploying

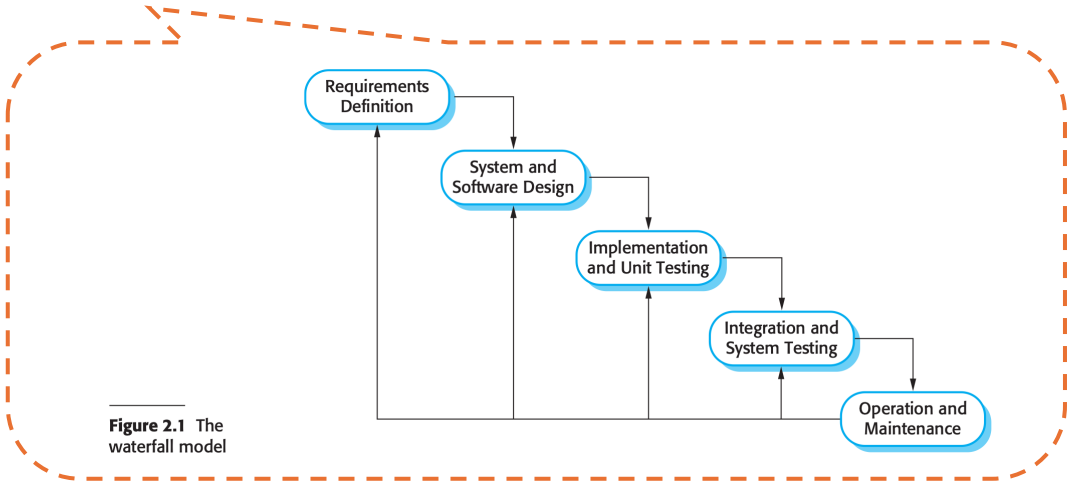
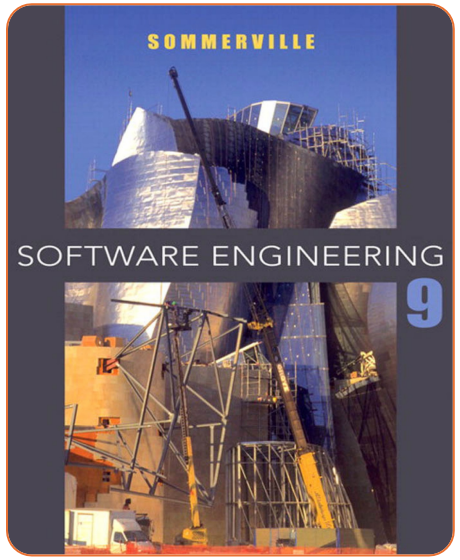


Figure 2.1 The waterfall model



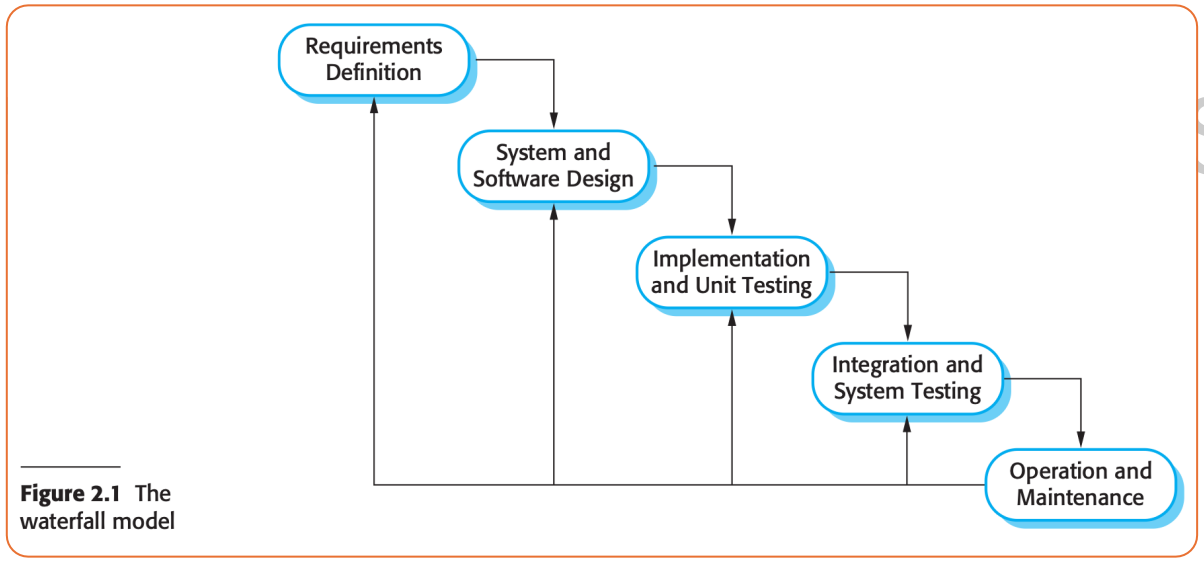


Figure 2.1 The waterfall model

SDLC)

1. Planning → 2. Analysis → 3. Design → 4. Coding → 5. Testing → 6. Deploying

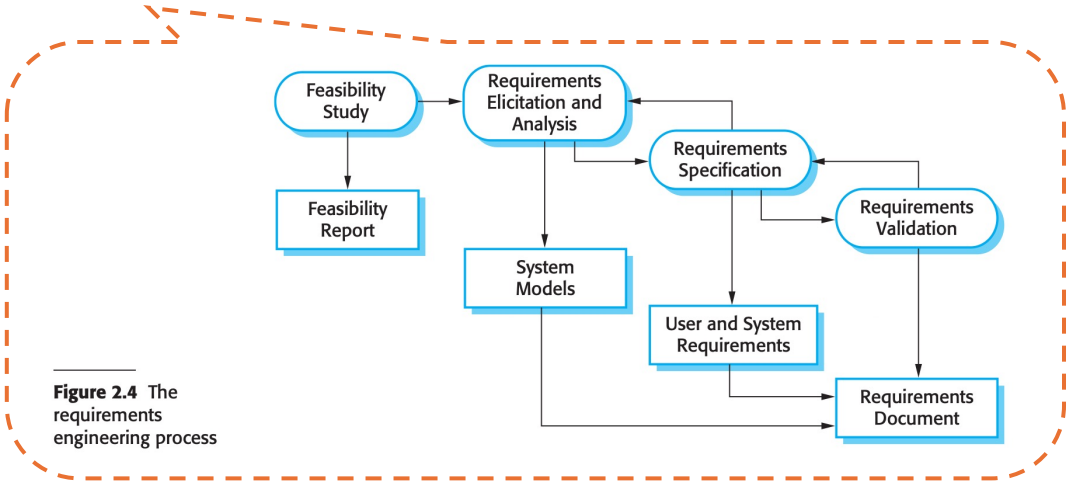
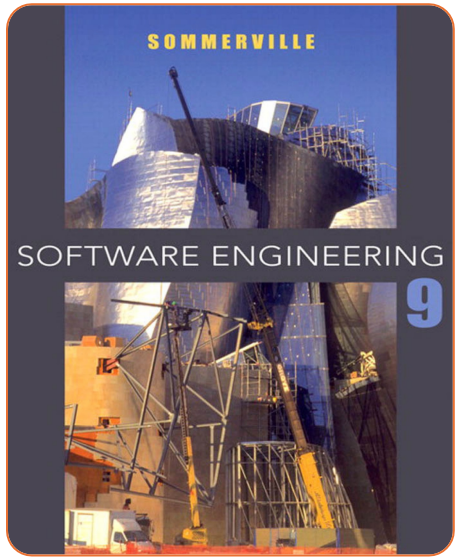


Figure 2.4 The requirements engineering process



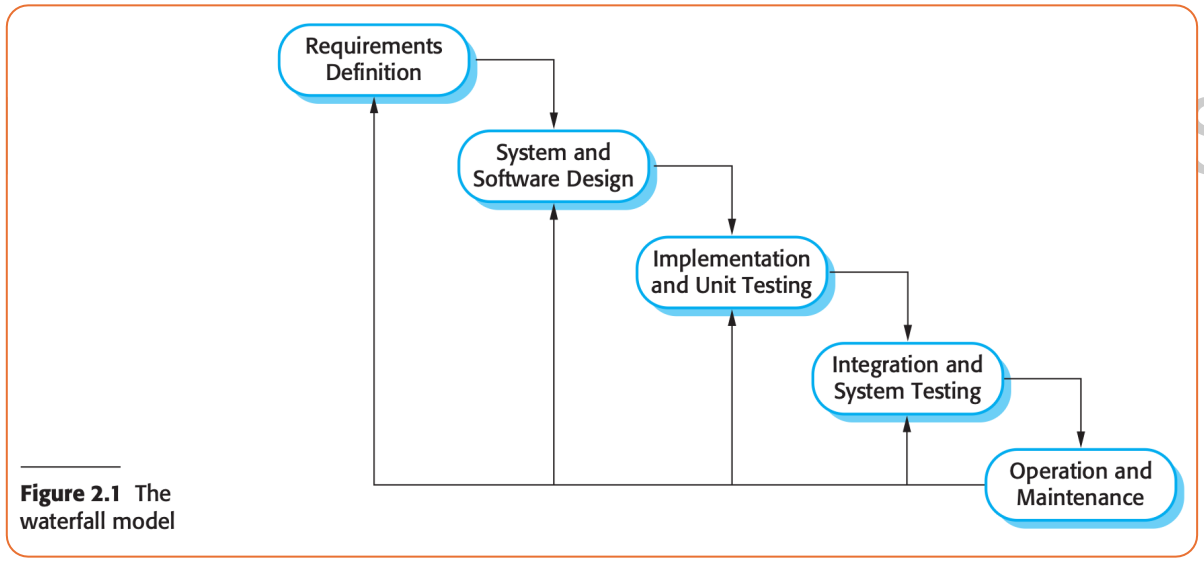


Figure 2.1 The waterfall model

SDLC)

1. Planning → 2. Analysis → 3. Design → 4. Coding → 5. Testing → 6. Deploying

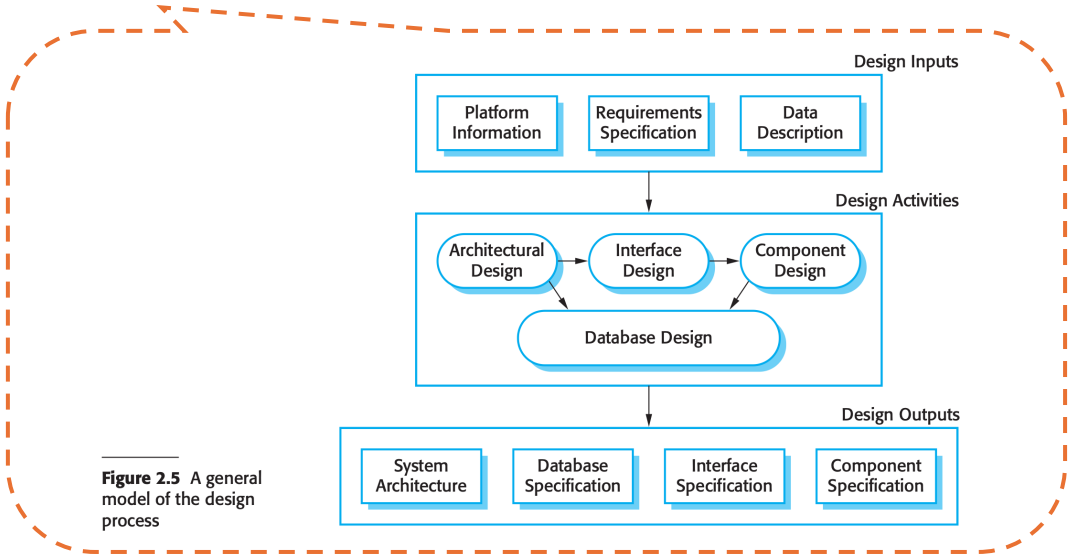
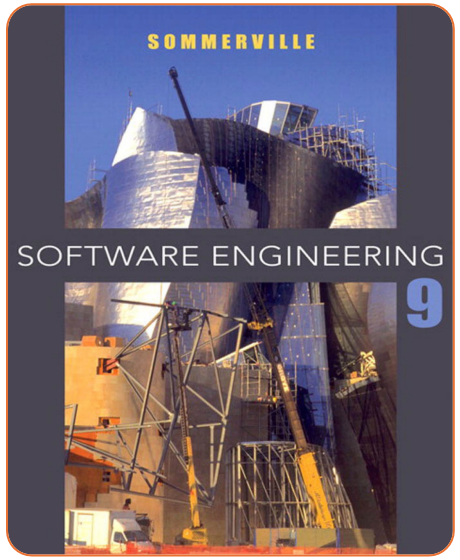


Figure 2.5 A general model of the design process



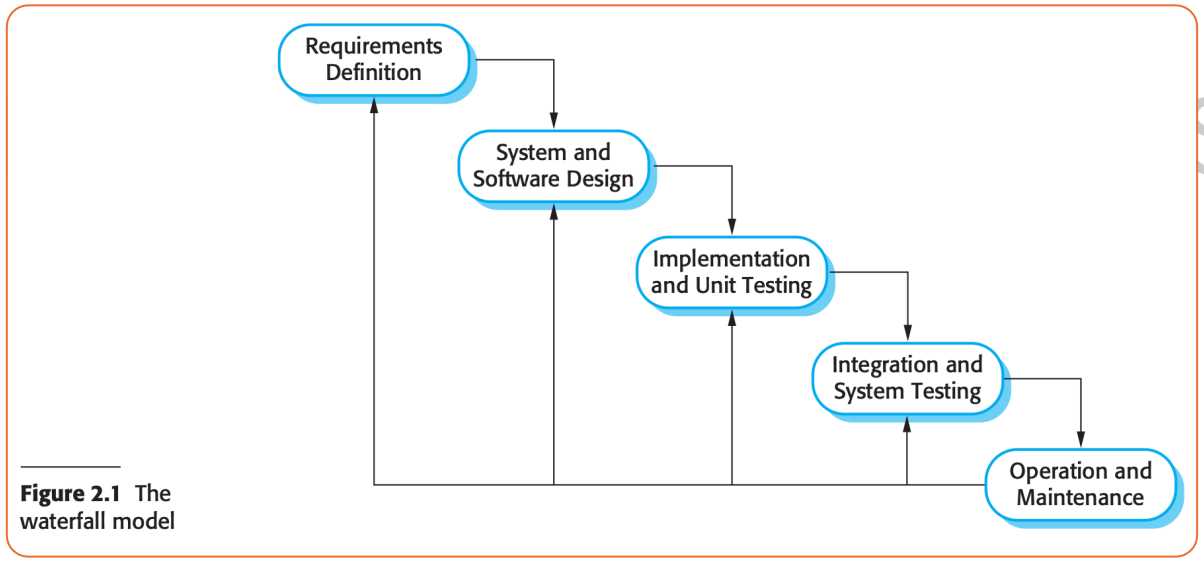
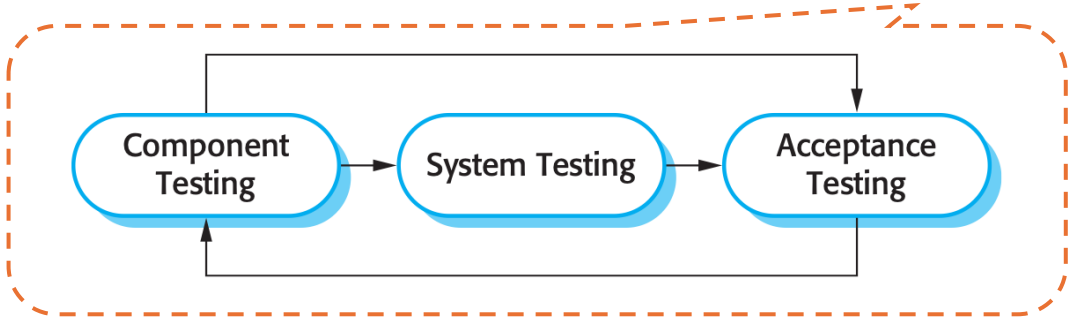


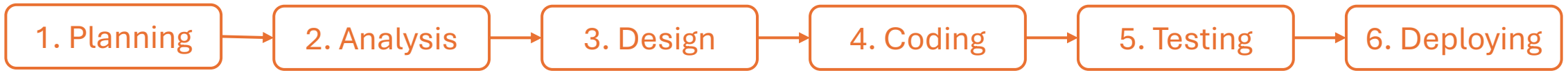
Figure 2.1 The waterfall model

SDLC)

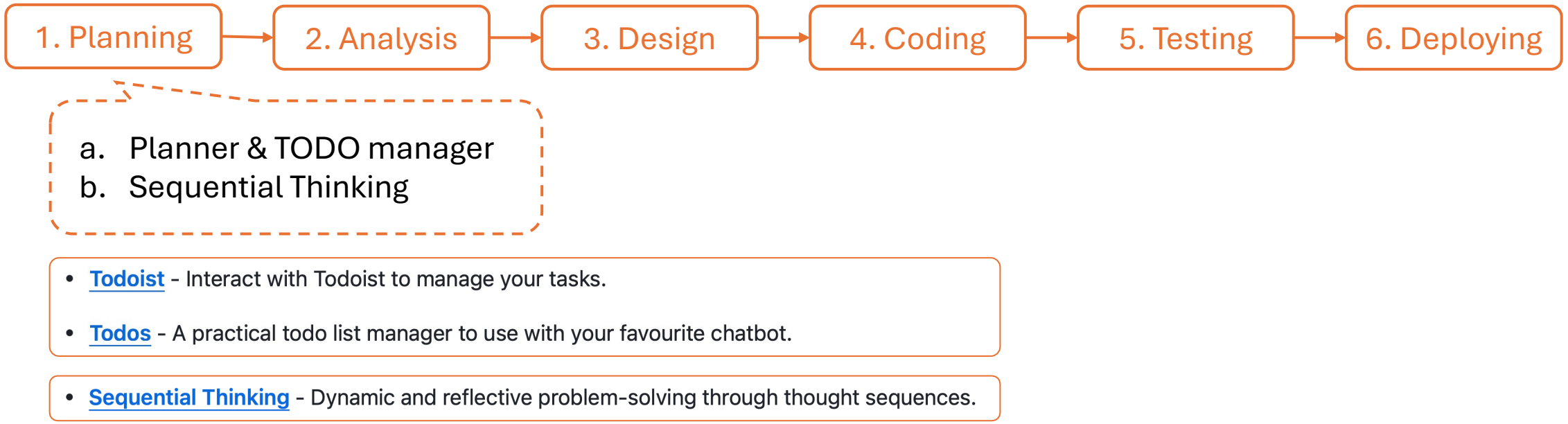
1. Planning → 2. Analysis → 3. Design → 4. Coding → 5. Testing → 6. Deploying



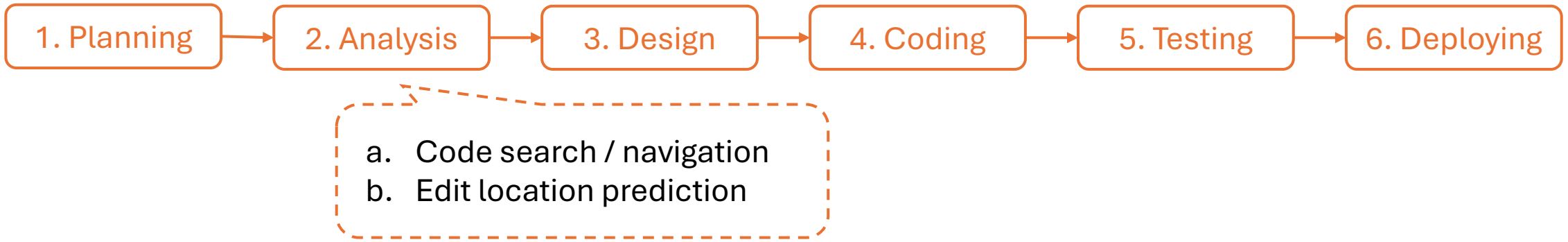
Agentic Framework for Software Development



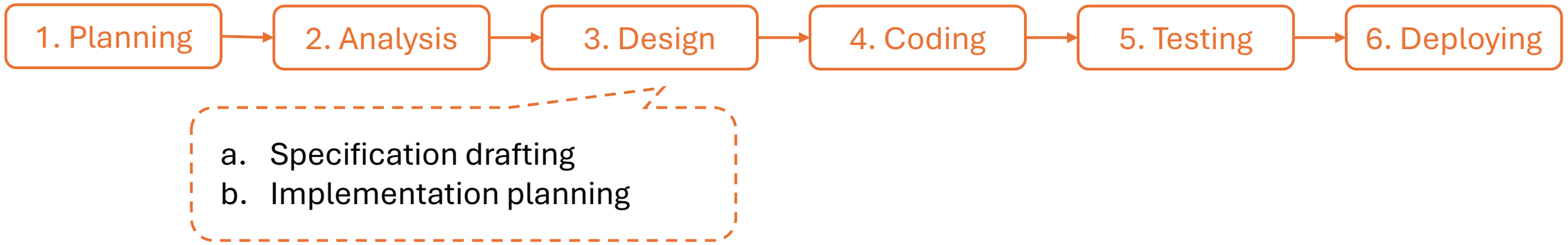
Agentic Framework for Software Development



Agentic Framework for Software Development



Agentic Framework for Software Development



Agentic Framework for Software Development

1. Planning → 2. Analysis → 3. Design → 4. Coding → 5. Testing → 6. Deploying

- **Filesystem** - Secure file operations with configurable access controls.
- **Language Server** - MCP Language Server helps MCP enabled clients navigate codebases giving them access to semantic tools like get definition, references, rename, and diagnosti
- **lean-lsp-mcp** - Interact with the Lean theorem prover via the Language Server Protocol.
- **Microsoft Learn Docs** - An MCP server that provides structured access to Microsoft's official documentation. Retrieves accurate, authoritative, and context-aware technical content for code generation, question answering, and workflow grounding.

a. File systems
b. Language servers (e.g., for Python, Rust, OCaml, ...)
c. Documentation indexed by RAG databases

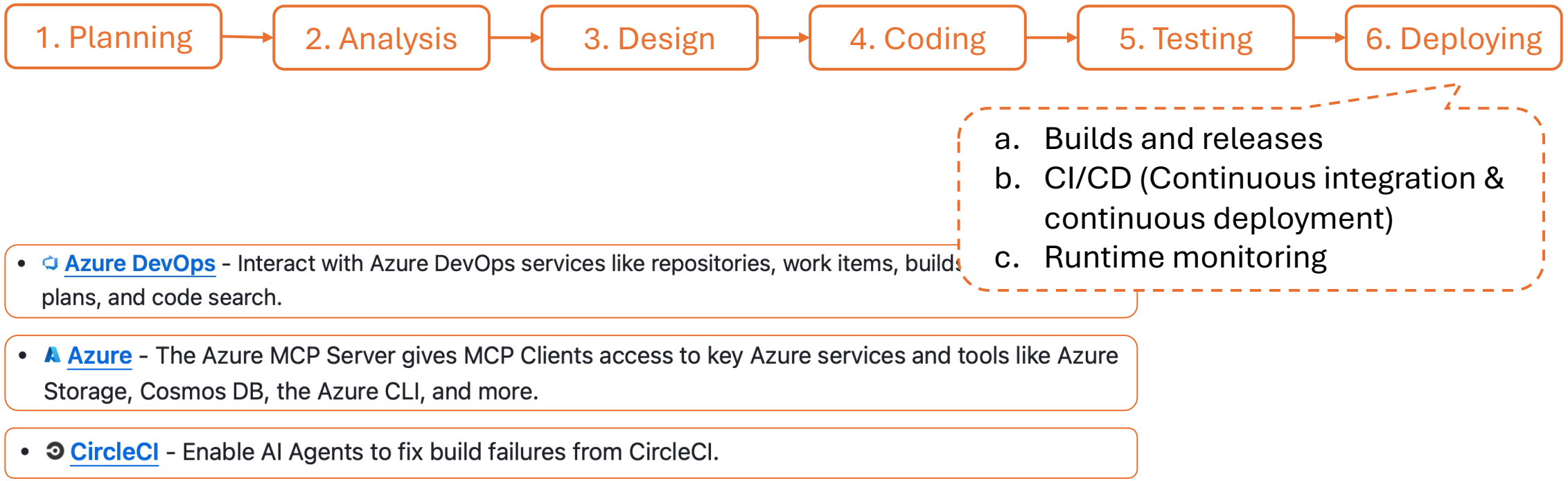
Agentic Framework for Software Development

1. Planning → 2. Analysis → 3. Design → 4. Coding → 5. Testing → 6. Deploying

a. Managing testing environments
b. Unit testing, regression testing
c. UI testing, accessibility testing
d. Vulnerability detection

- **Azure DevOps** - Interact with Azure DevOps services like repositories, work items, build plans, and code search.
- **BrowserStack** - Access BrowserStack's Test Platform to debug, write and fix tests, do accessibility testing and more.
- **Debugg.AI** - Zero-Config, Fully AI-Managed End-to-End Testing for any code gen platform via Debugg.AI remote browsing test agents.
- **CrowdStrike Falcon** - Connects AI agents with the CrowdStrike Falcon platform for intelligent security analysis, providing programmatic access to detections, incidents, behaviors, threat intelligence, hosts, vulnerabilities, and identity protection capabilities.

Agentic Framework for Software Development



Tools Related to Software Development

- Sequential thinking
  - Planning, deliberate thinking
- File systems (FS)
  - Creating, removing, editing programs
  - Drafting test cases and setting up testing environment
- Retrieval augmented generation (RAG) databases
  - Context management
  - Documentations, examples, code snippets

Sequential Thinking Tool

- Externalized Chain-of-thought
- The Sequential Thinking tool is designed for:
  - Breaking down complex problems into steps
  - Planning and design with room for revision
  - Analysis that might need course correction
  - Problems where the full scope might not be clear initially
  - Tasks that need to maintain context over multiple steps
  - Situations where irrelevant information needs to be filtered out

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

To-dos 7

- Create main package structure and exports in __init__.py
- Implement base Tool class and tool registry system
- Create LLM interface and JSON parsing system
- Implement core Agent class with plan/act loop
- Create CLI interface for waa command
- Set up logging system
- Create tool implementations (todo, fs, server, tests)

Now let me start implementing the high-level structure. First, I'll create the main package structure:

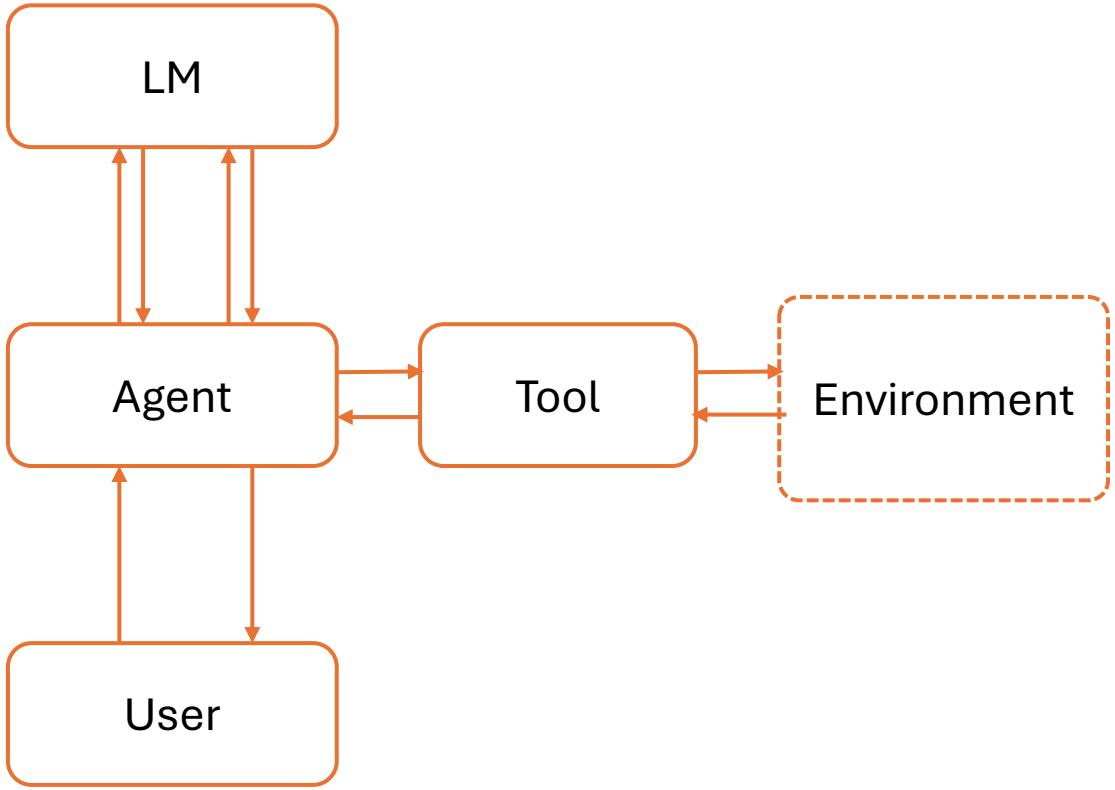
__init__.py +17 -1

1 of 7 Done View all

- ~~Create main package structure and exports in __init__.py~~
- Implement base Tool class and tool registry system

Sequential Thinking Tool

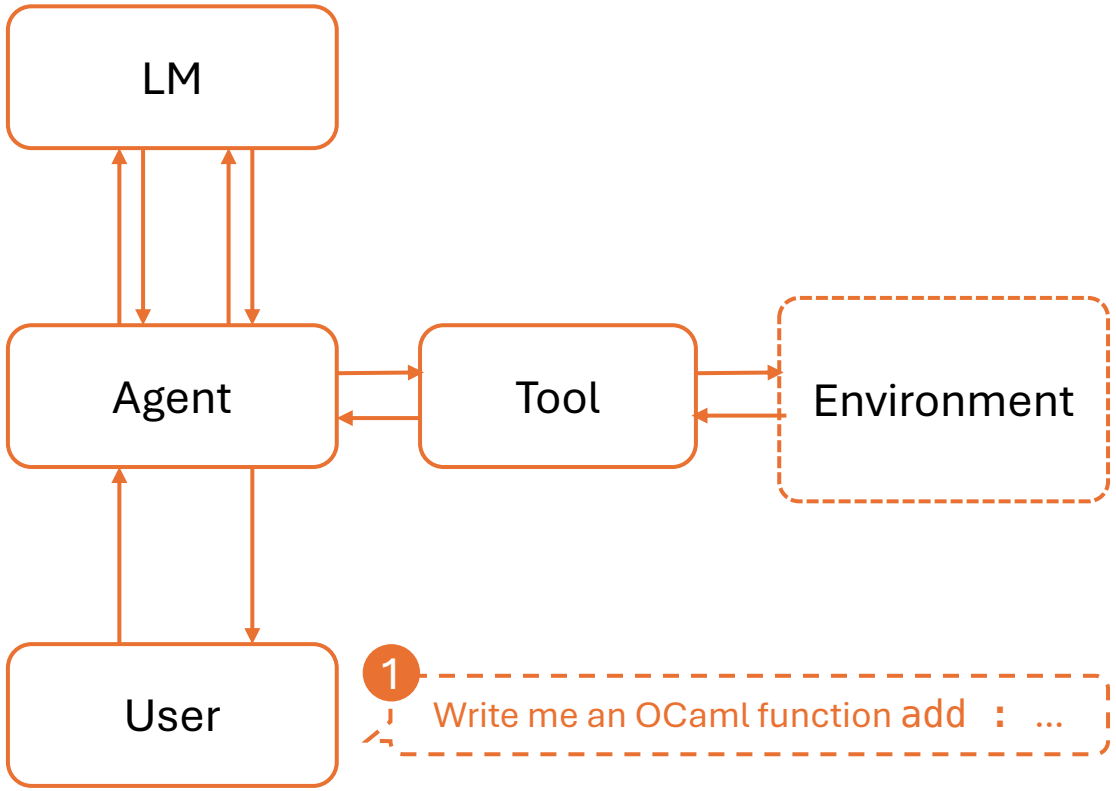
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you intend to add another step
- `thoughtNumber` (int ≥1)
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need



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

Sequential Thinking Tool

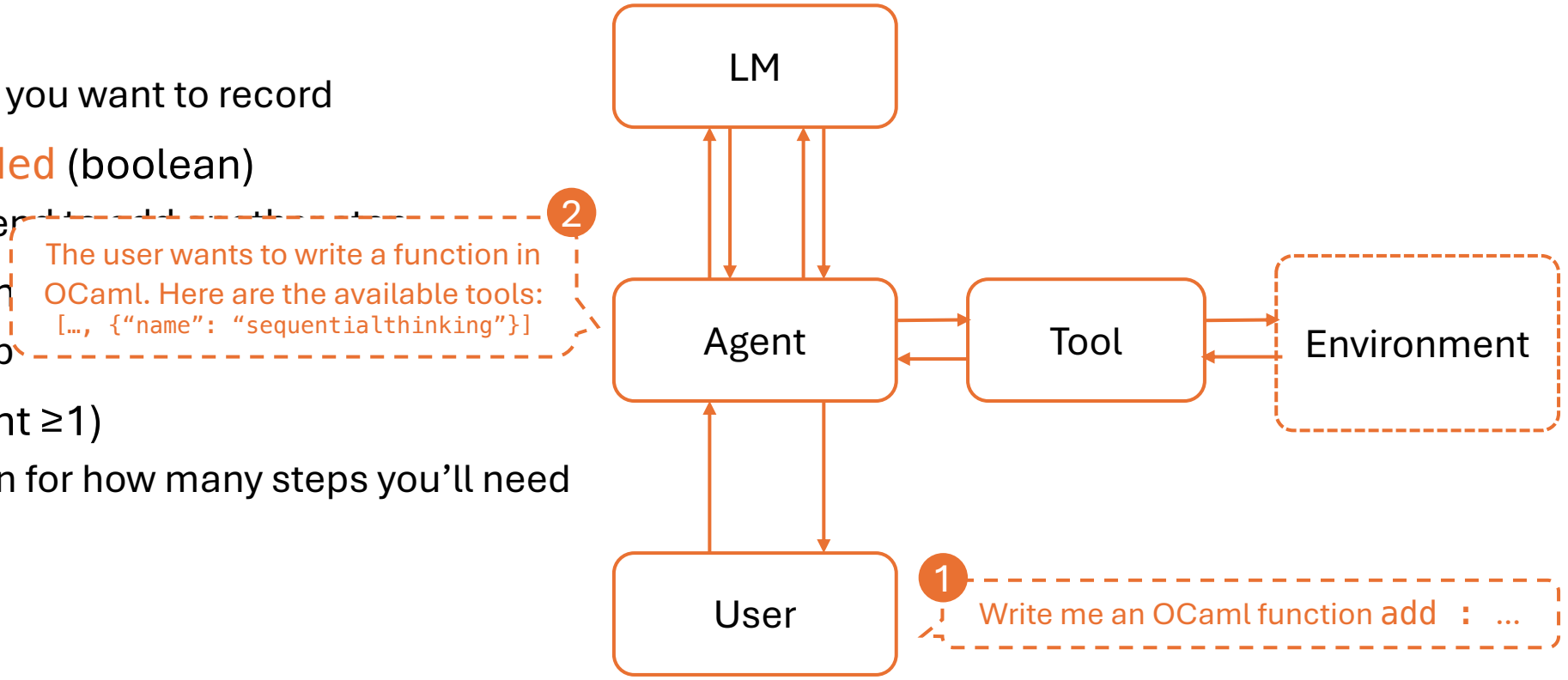
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you intend to add another step
- `thoughtNumber` (int ≥1)
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need



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

Sequential Thinking Tool

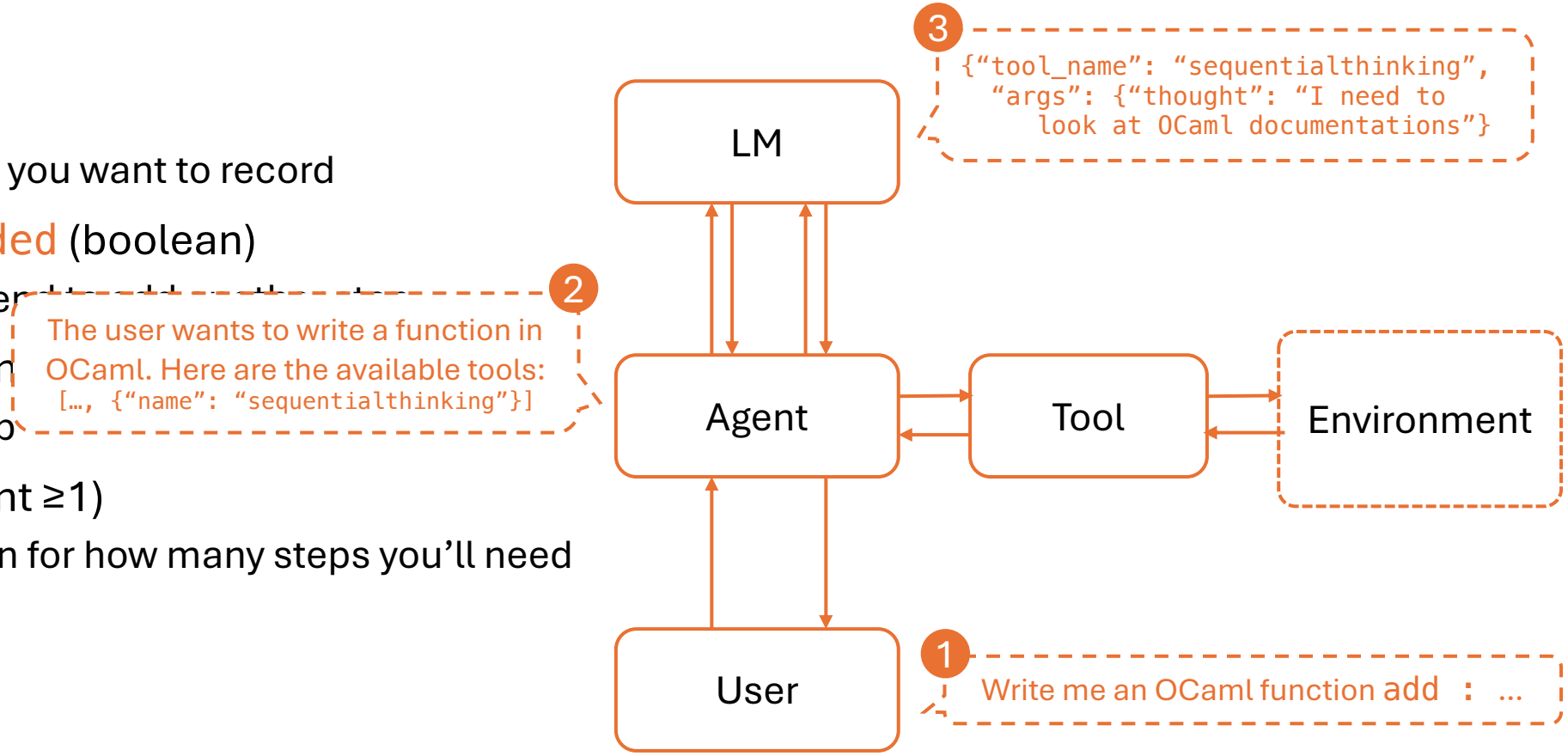
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you inter[illegible]
- `thoughtNumber` (in[illegible]
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need



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

Sequential Thinking Tool

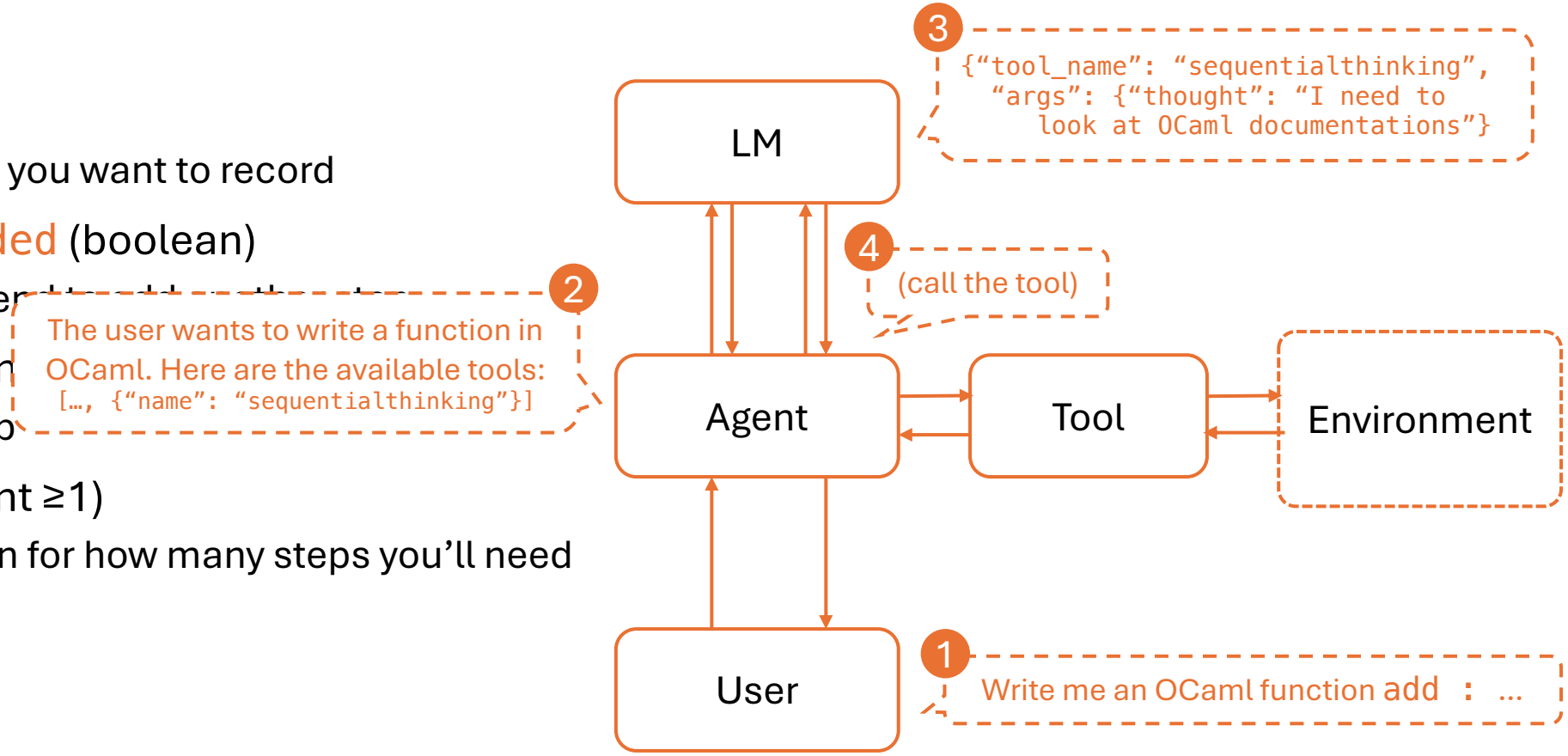
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you inter
- `thoughtNumber` (in
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need



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

Sequential Thinking Tool

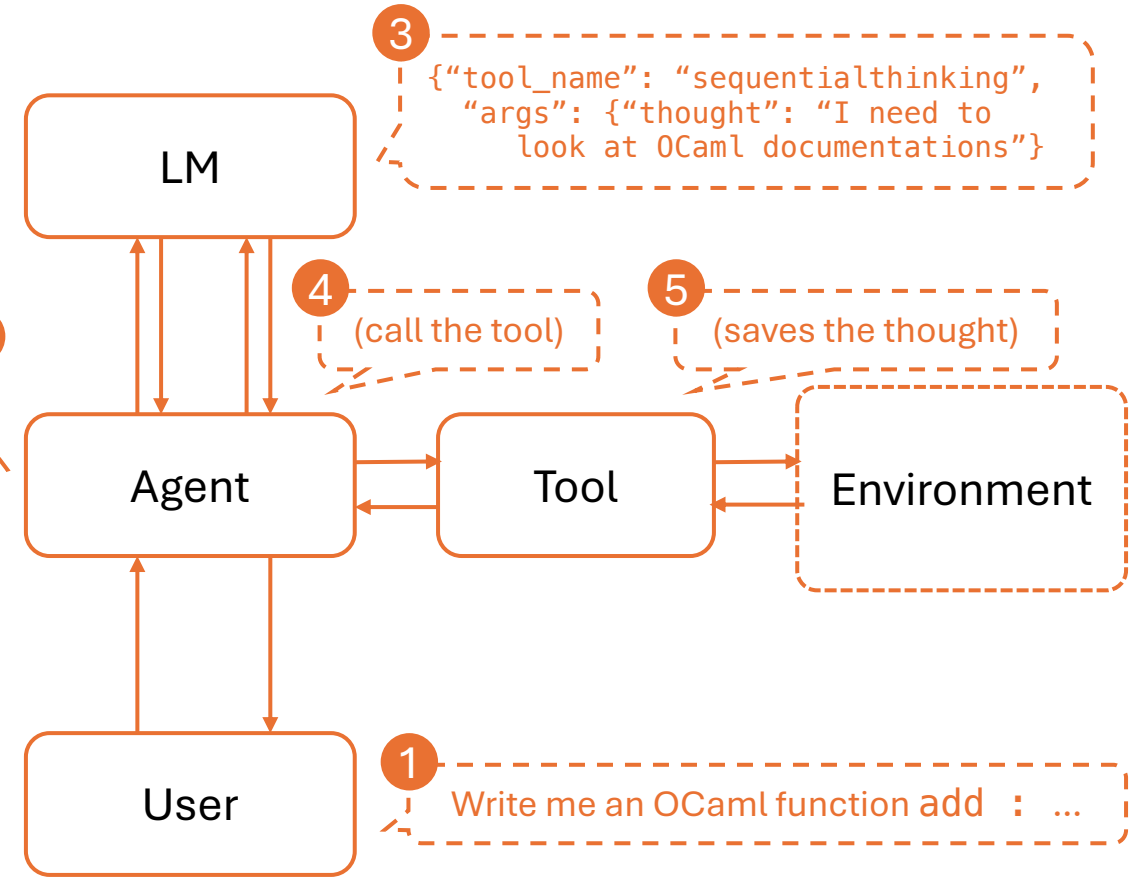
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you inter
- `thoughtNumber` (in
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need



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

Sequential Thinking Tool

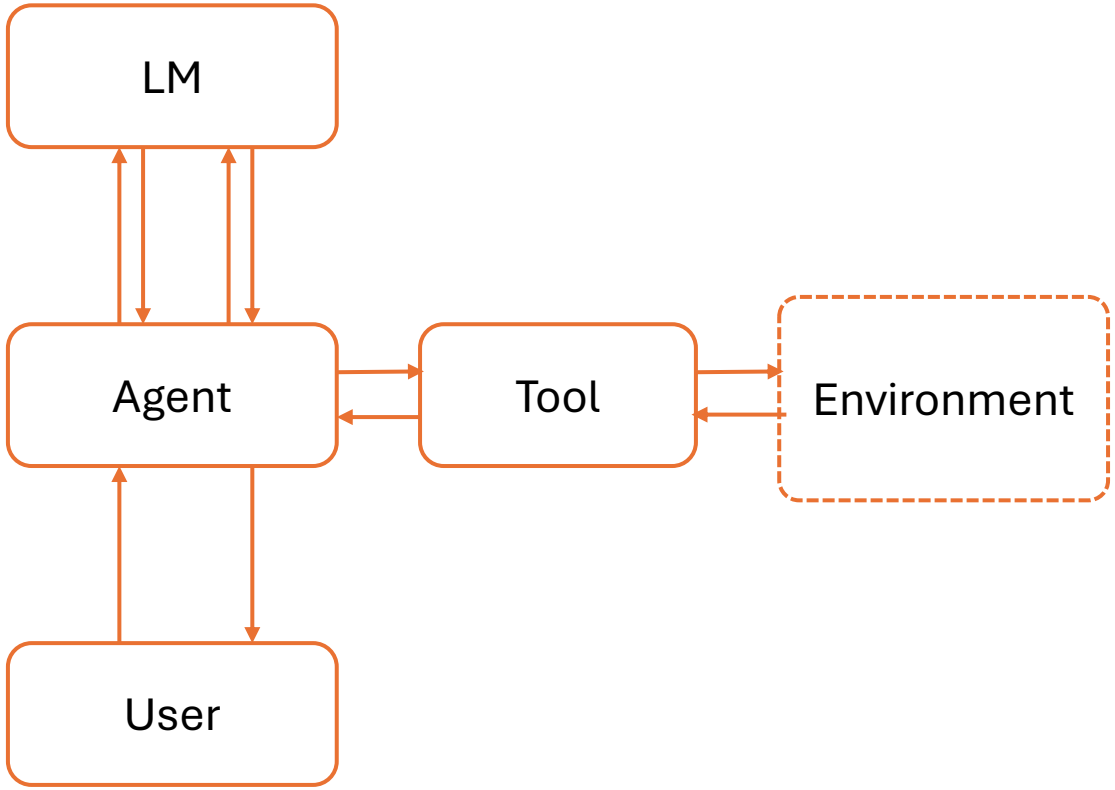
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you inter
- `thoughtNumber` (in
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need



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

Sequential Thinking Tool

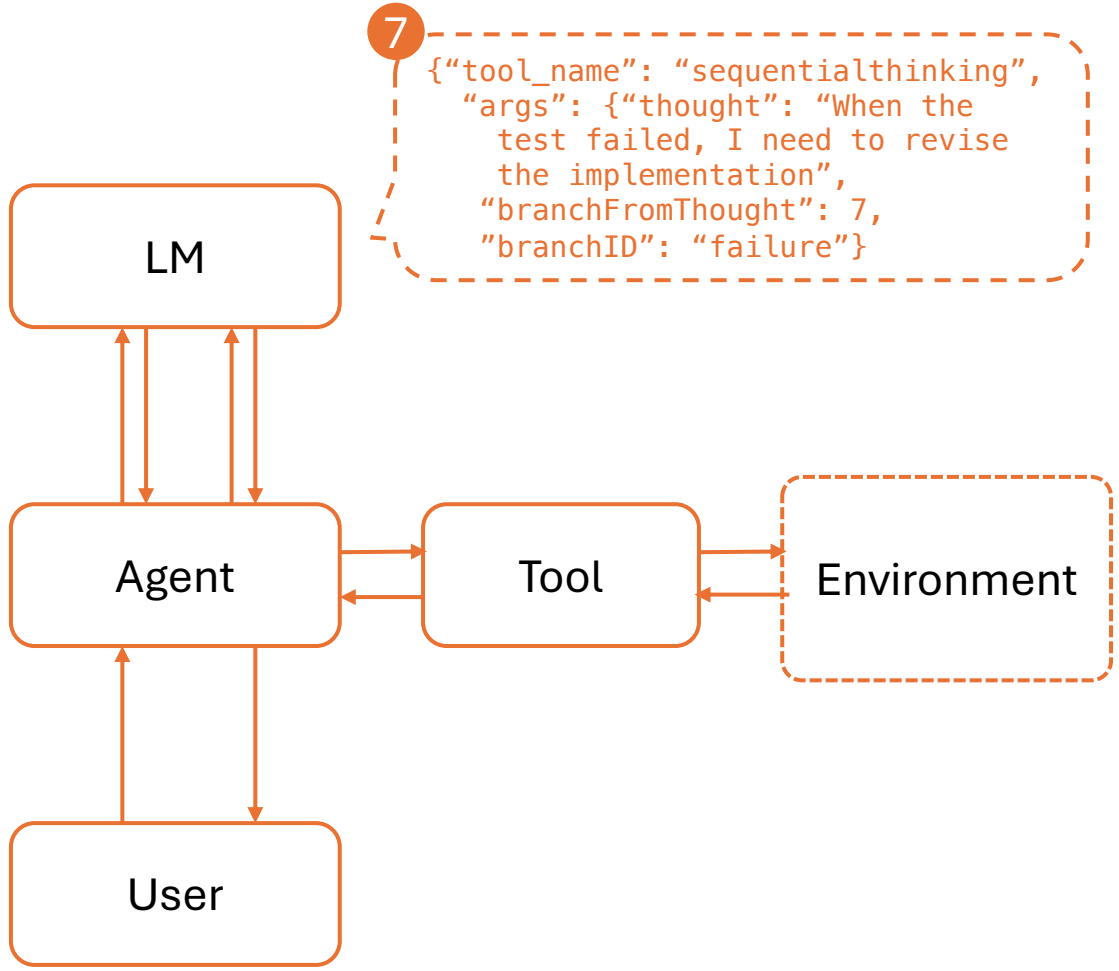
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you intend to add another step
- `thoughtNumber` (int ≥1)
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need
- `branchFromThought` (int ≥1)
-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



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

Sequential Thinking Tool

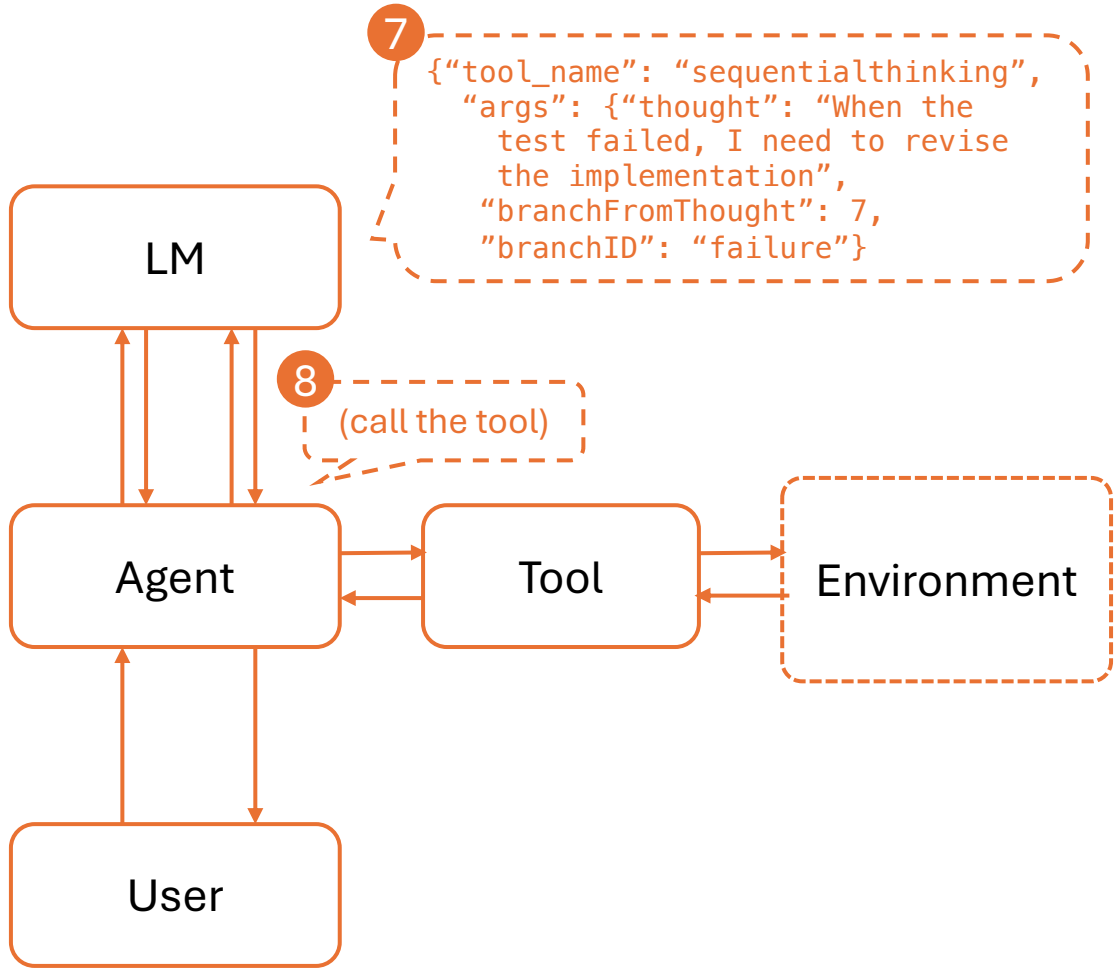
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you intend to add another step
- `thoughtNumber` (int ≥1)
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need
- `branchFromThought` (int ≥1)
-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



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

Sequential Thinking Tool

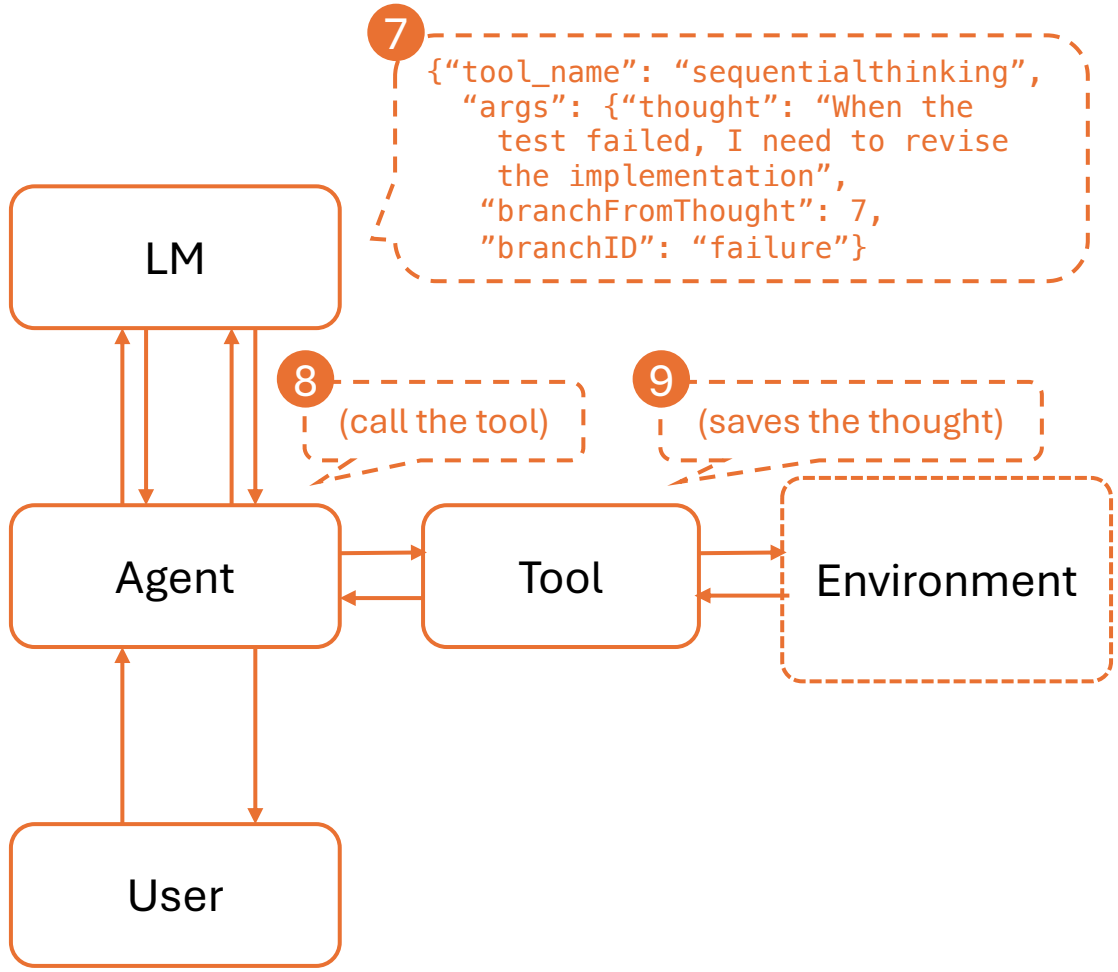
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you intend to add another step
- `thoughtNumber` (int ≥1)
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need
- `branchFromThought` (int ≥1)
-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



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

Sequential Thinking Tool

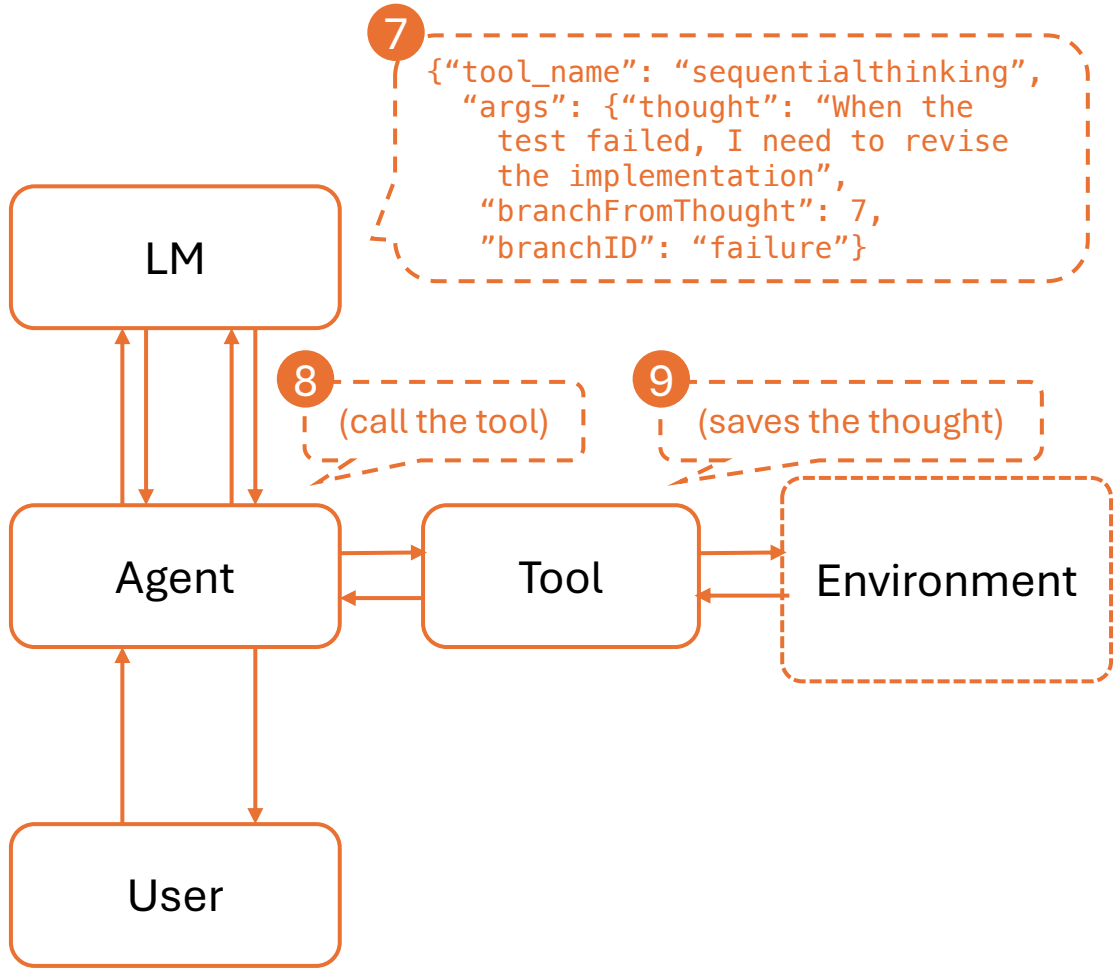
- `thought` (string)
  - the current step you want to record
- `nextThoughtNeeded` (boolean)
  - whether you intend to add another step
- `thoughtNumber` (int ≥1)
  - index of this step
- `totalThoughts` (int ≥1)
  - your *current* plan for how many steps you'll need
- `branchFromThought` (int ≥1)
-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



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

Sequential Thinking Tool

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



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

To-dos 7

- Create main package structure and exports in __init__.py
- Implement base Tool class and tool registry system
- Create LLM interface and JSON parsing system
- Implement core Agent class with plan/act loop
- Create CLI interface for waa command
- Set up logging system
- Create tool implementations (todo, fs, server, tests)

Now let me start implementing the high-level structure. First, I'll create the main package structure:

__init__.py +17 -1

1 of 7 Done View all

- ~~Create main package structure and exports in __init__.py~~
- Implement base Tool class and tool registry system

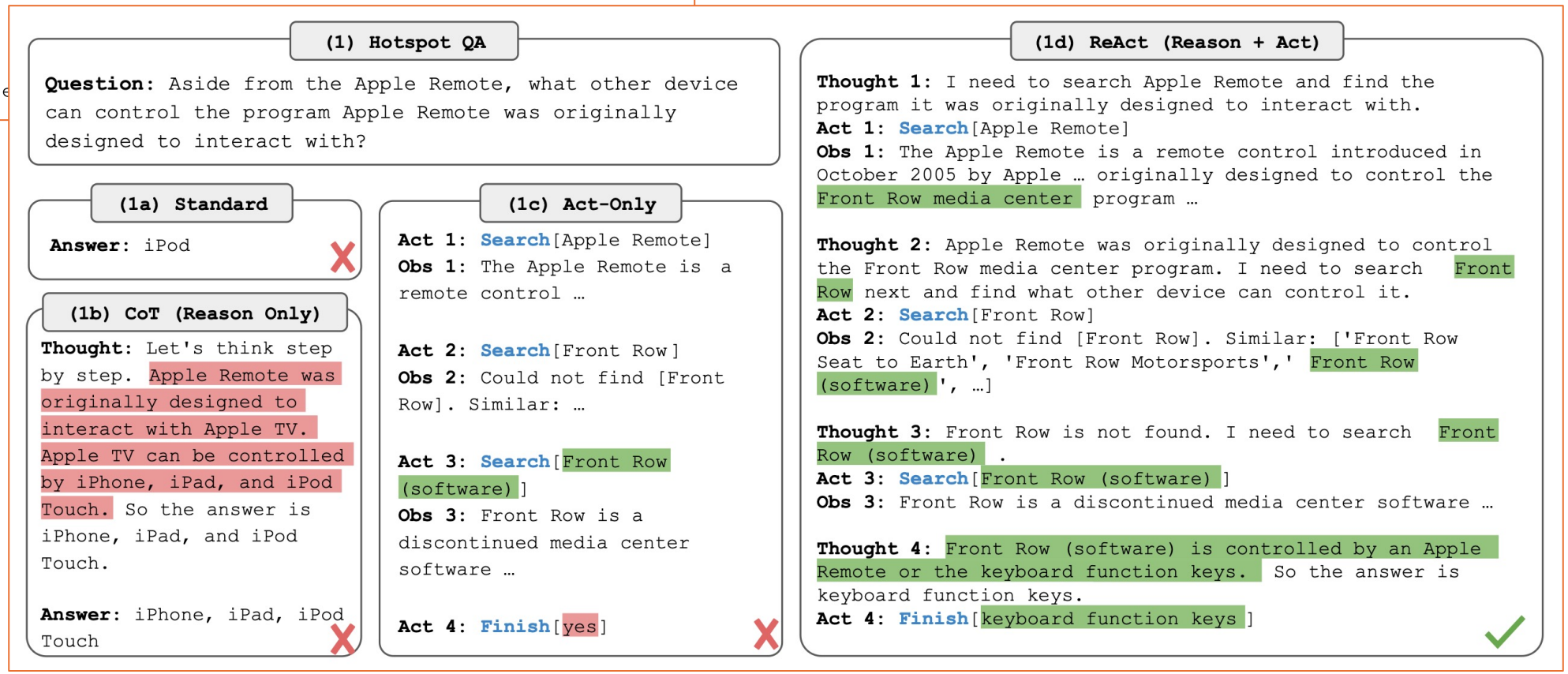
REACT: SYNERGIZING REASONING AND ACTING IN LANGUAGE MODELS

Shunyu Yao[*,1], Jeffrey Zhao[2], Dian Yu[2], Nan Du[2], Izhak Shafran[2], Karthik Narasimhan[1], Yuan Cao[2]

[1]Department of Computer Science, Princeton University
[2]Google Research, Brain team
[1]{shunyuy,karthikn}@princeton.edu
[2]{jeffreyzhao,dianyu,dunan,izhak,yuancao}@google.com

REACT: SYNERGIZING REASONING AND ACTING IN LANGUAGE MODELS

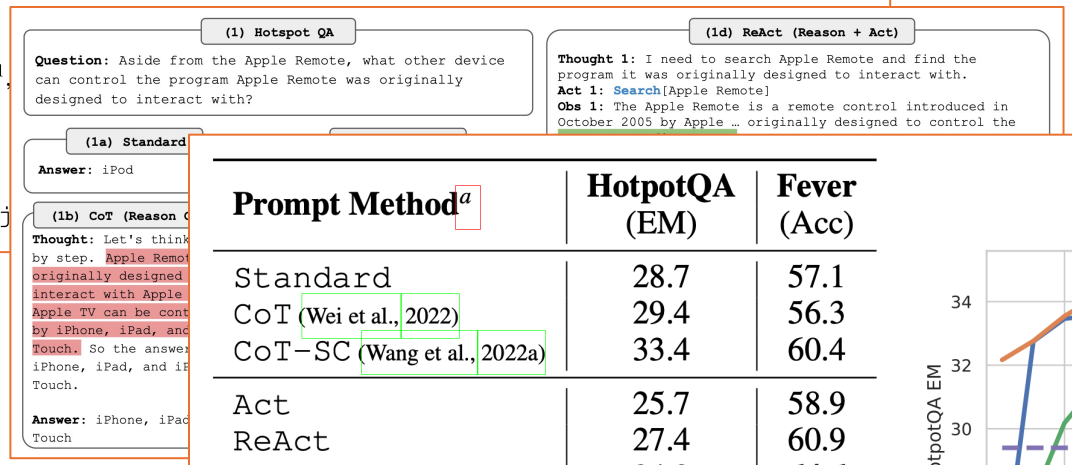
Shunyu Yao*[,1], Jeffrey Zhao[2], Dian Yu[2], Nan Du[2], Izhak Shafran[2], Karthik Narasimhan[1], Yuan Cao[2]



ReACT: Synergizing Reasoning and Acting in Language Models, Yao et. al., ICLR 2023

REACT: SYNERGIZING REASONING AND ACTING IN LANGUAGE MODELS

Shunyu Yao*,1,



Prompt Method[a]	HotpotQA (EM)	Fever (Acc)
Standard	28.7	57.1
CoT (Wei et al., 2022)	29.4	56.3
CoT-SC (Wang et al., 2022a)	33.4	60.4
Act	25.7	58.9
ReAct	27.4	60.9
CoT-SC → ReAct	34.2	**64.6**
ReAct → CoT-SC	**35.1**	62.0
Supervised SoTA[b]	67.5	89.5

Table 1: PaLM-540B prompting results on HotpotQA and Fever.

[a] HotpotQA EM is 27.1, 28.9, 33.8 for Standard, CoT, CoT-SC in Wang et al. (2022b).

[b] (Zhu et al., 2021; Lewis et al., 2020)

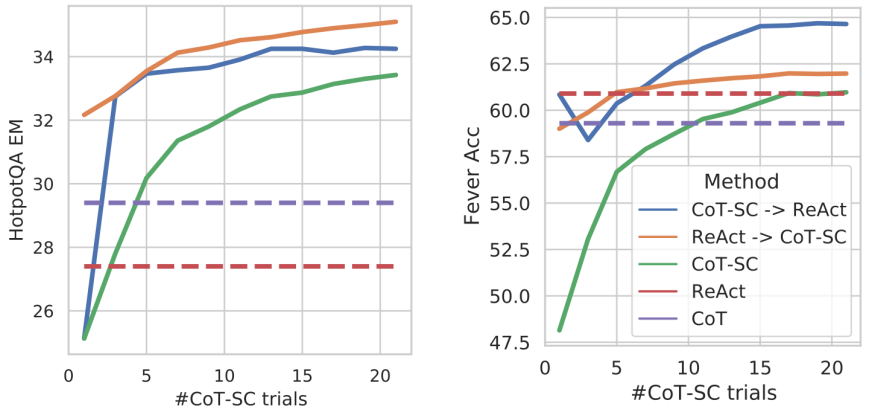


Figure 2: PaLM-540B prompting results with respect to number of CoT-SC samples used.

File System (FS) for Code Implementation

- Create, read, write, or edit files
  - Managing project folder structures
  - Put implementation in relevant files, write test cases, etc.
- Search for files
  - Good when we do not know what exact files to work on
  - Often substituted by command line tools such as “grep”

Can you look at core/scallop-fir/docs/ and help me implement a type_checker

I'll help you implement a type checker for the Scallop language. Let me first explore the documentation and understand the current structure of the project.

Listed docs

Read grammar.md

Searched What is the current

Let me explore the existing

Read test_type_inference.rs

Read test_type_bounds.rs

Searched Where are type def

Let me explore the source c

Listed src

Read lib.rs

Grepped type.*check|TypeChecker|type_infere

I can see that the type checking functionalit

s . Let me explore the AST types to understa

Let me explore more of the type system structure:

Read mod.rs

Read type_decl.rs

Read named_type.rs

Now let me create a comprehensive plan for implementing the type checker:

1 of 9 Done

~~Explore AST structure and understand type system components~~

Create the analysis module with type checking infrastructure

Implemer

Now let me create the analysis module with the type checking infrastructure:

mod.rs (new) +3 •

type_inference.rs (new) +510 •

Now let me update the lib.rs to enable t

lib.rs +1 -1 •

Let me check for any compilation errors

No linter errors found in analysis

I need to fix the generic type parameter structure. Let me check the actual structure:

type_inference.rs +1 -1 •

Now let me check the ADT declaration structure:

Read adt_decl.rs

I need to fix the ADT checking code:

type_inference.rs +7 -2 •

Let me check the alias type declaration structure:

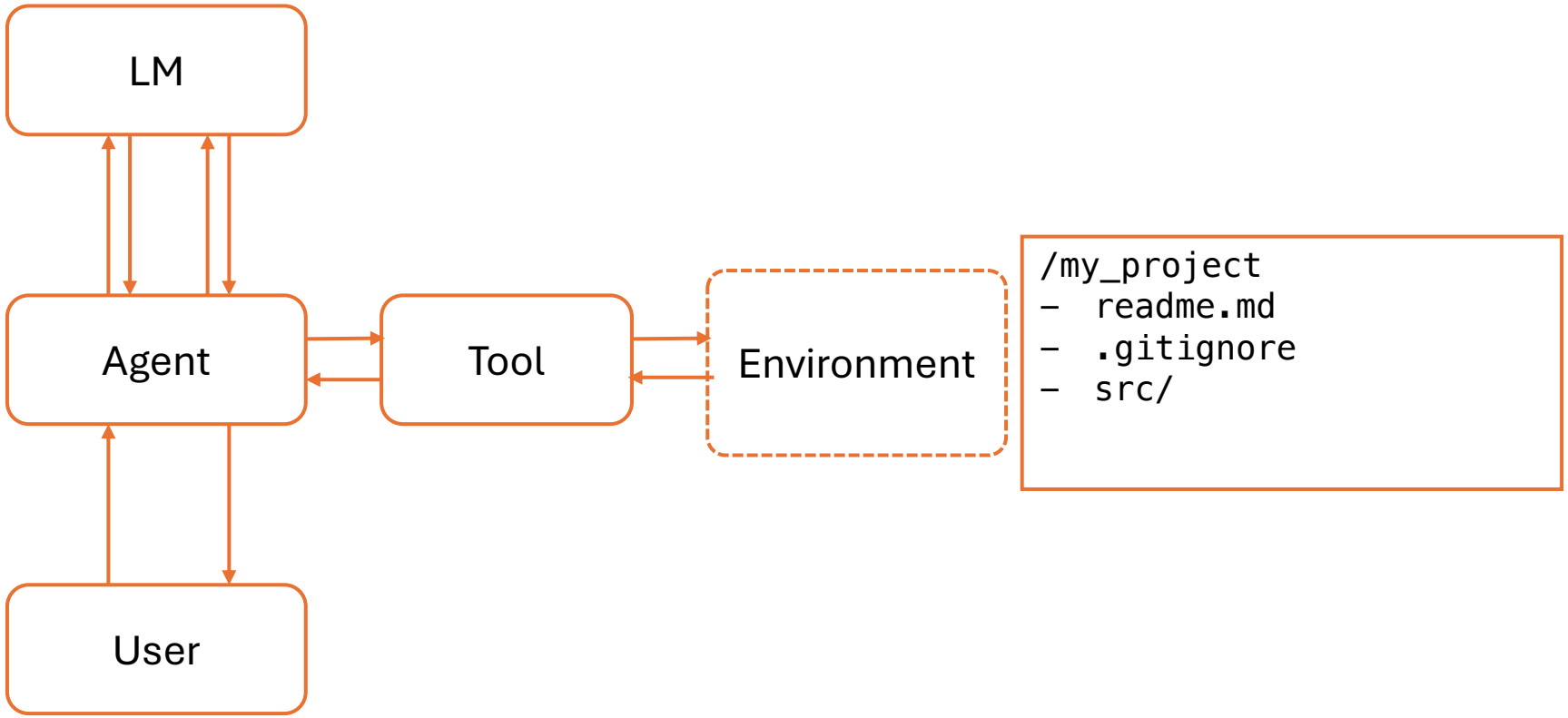
Read alias_type_decl.rs

I need to fix the alias type declaration checking:

type_inference.rs +7 -1 •

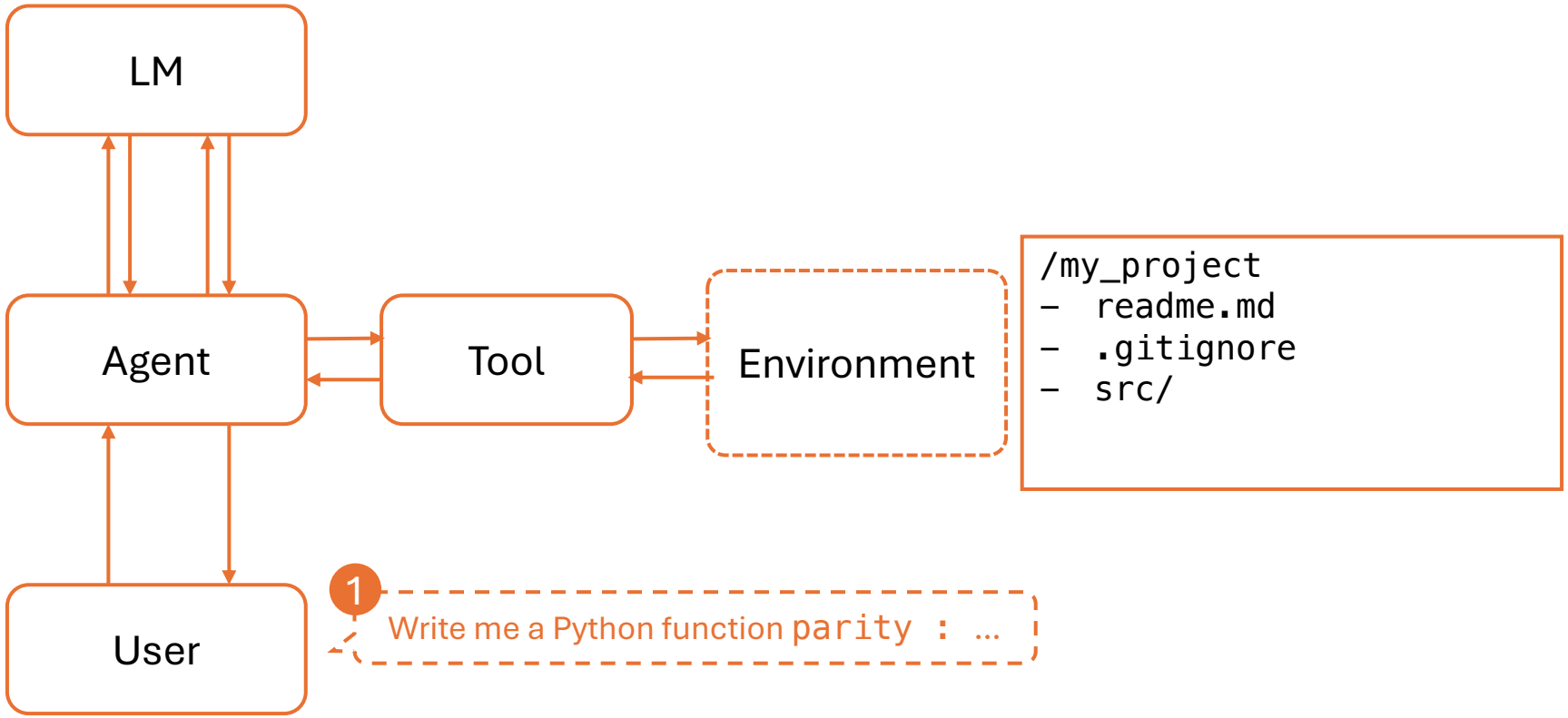
File System (FS) for Code Implementation

- Create, read, write, or edit files



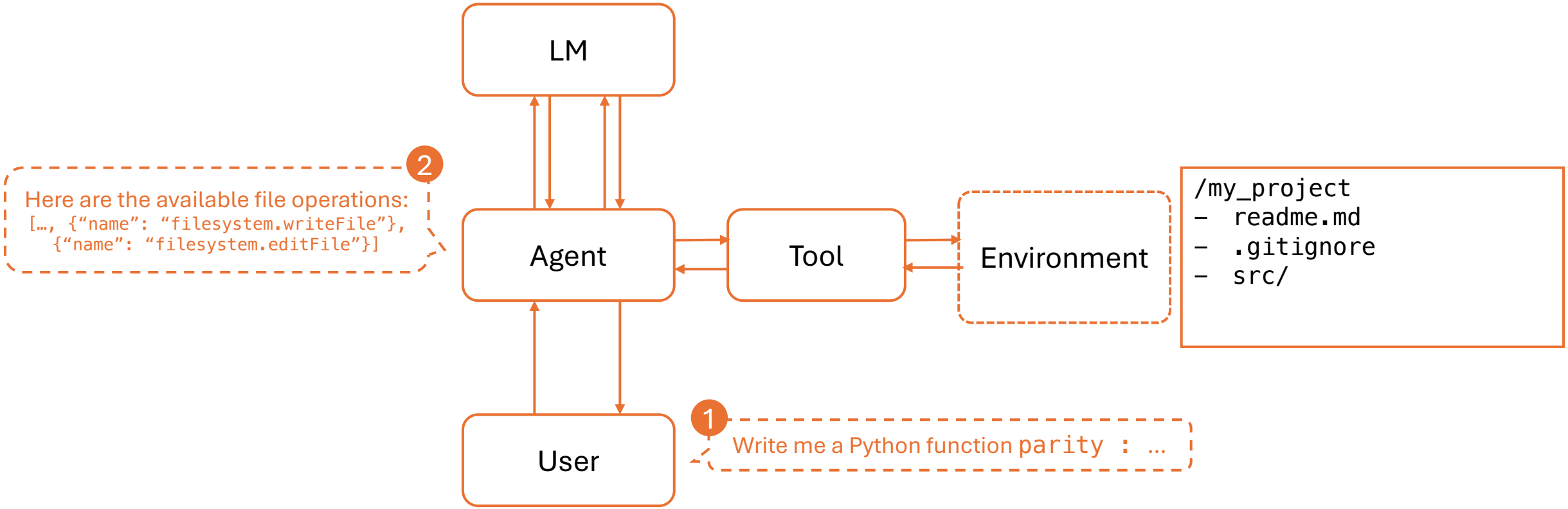
File System (FS) for Code Implementation

- Create, read, write, or edit files



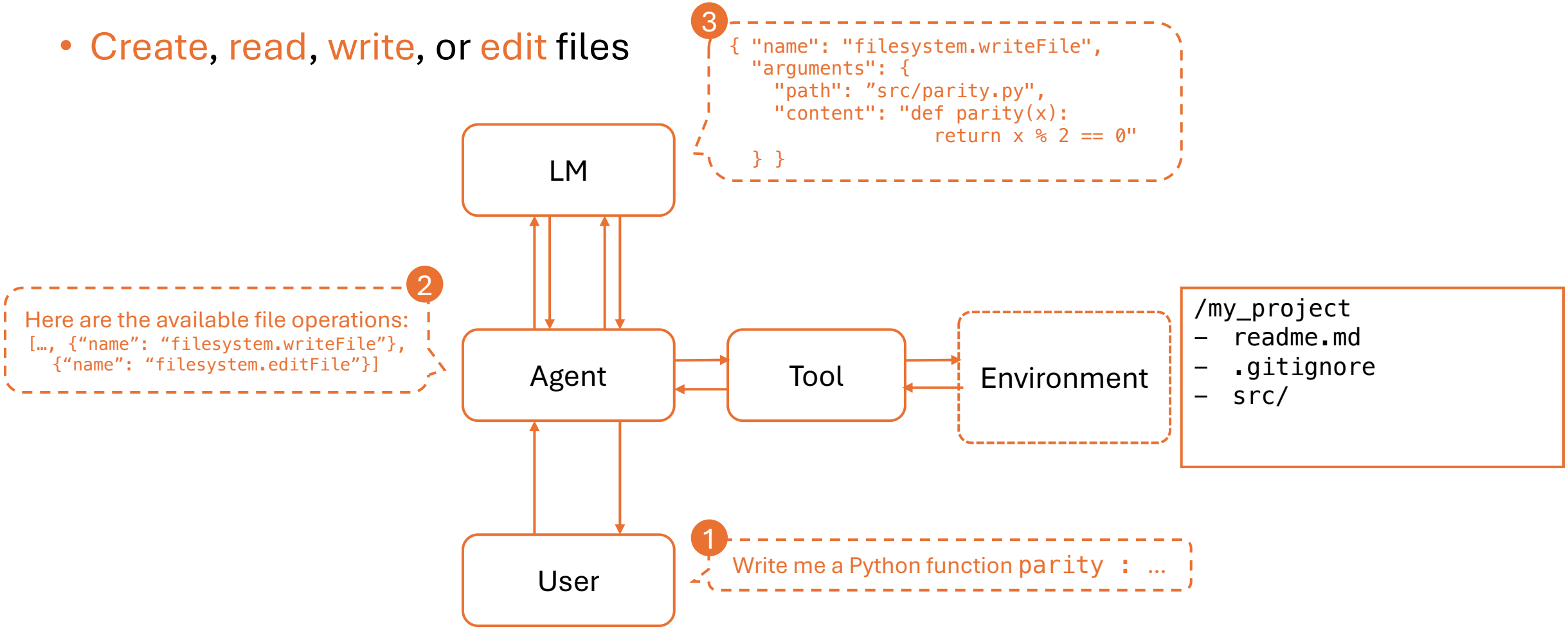
File System (FS) for Code Implementation

- Create, read, write, or edit files



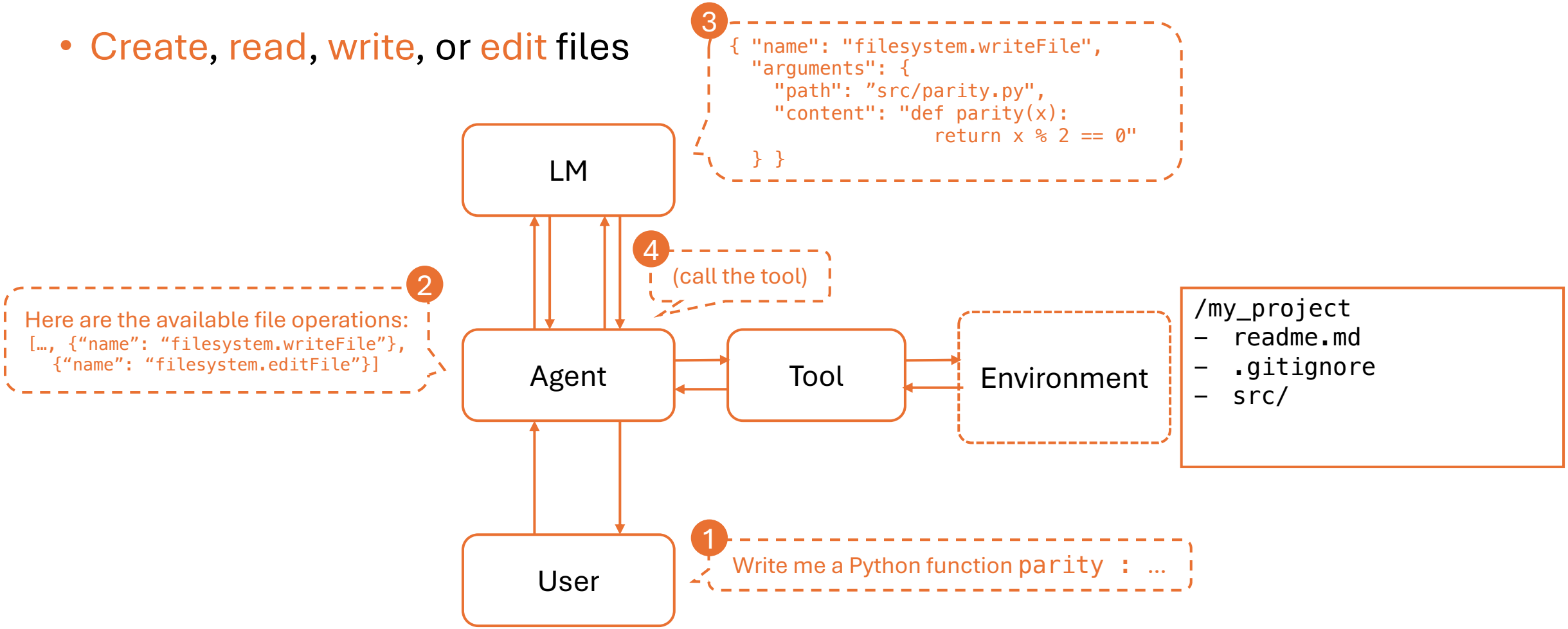
File System (FS) for Code Implementation

- Create, read, write, or edit files



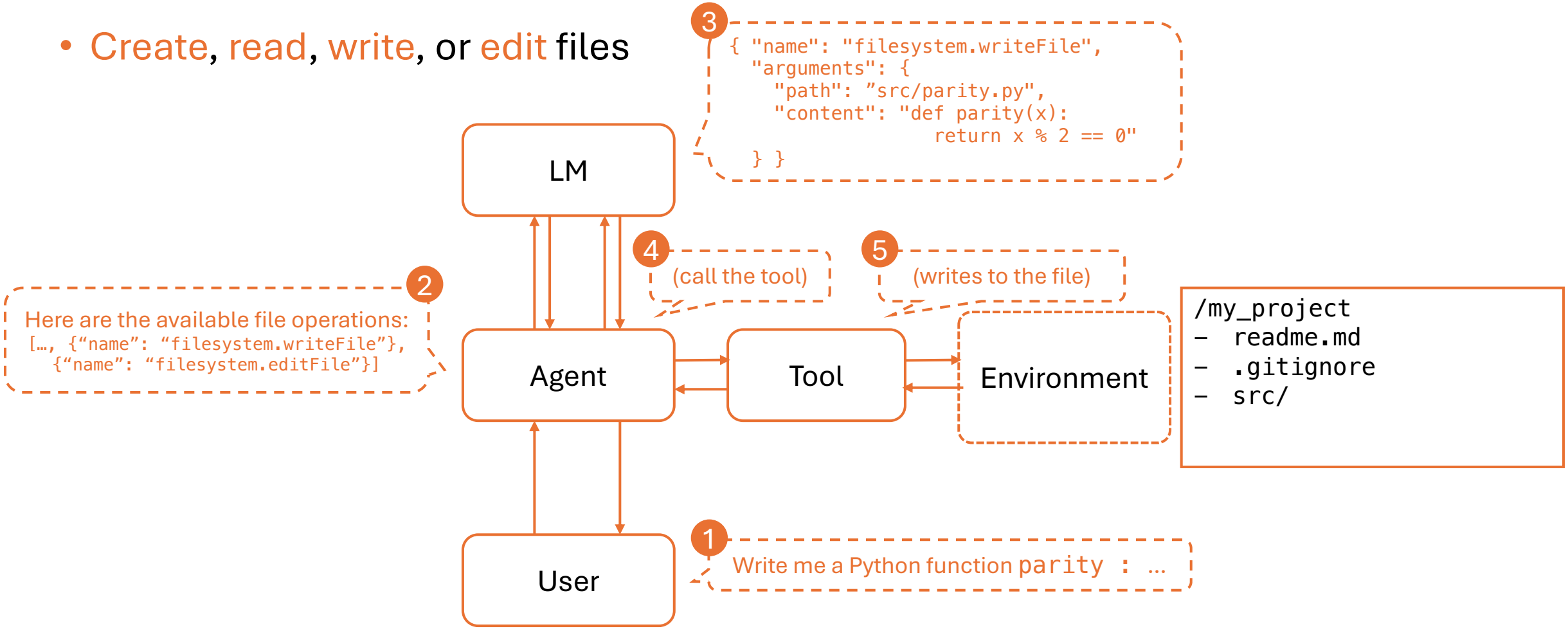
File System (FS) for Code Implementation

- Create, read, write, or edit files



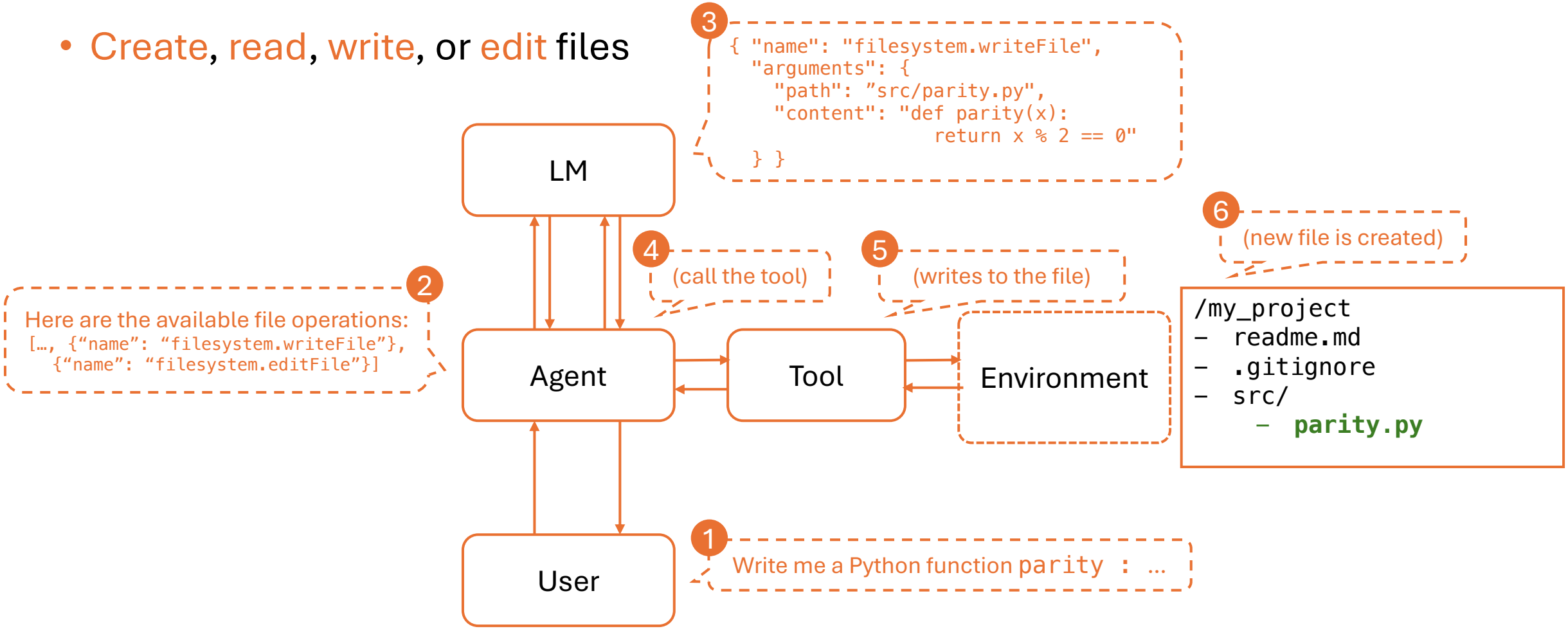
File System (FS) for Code Implementation

- Create, read, write, or edit files



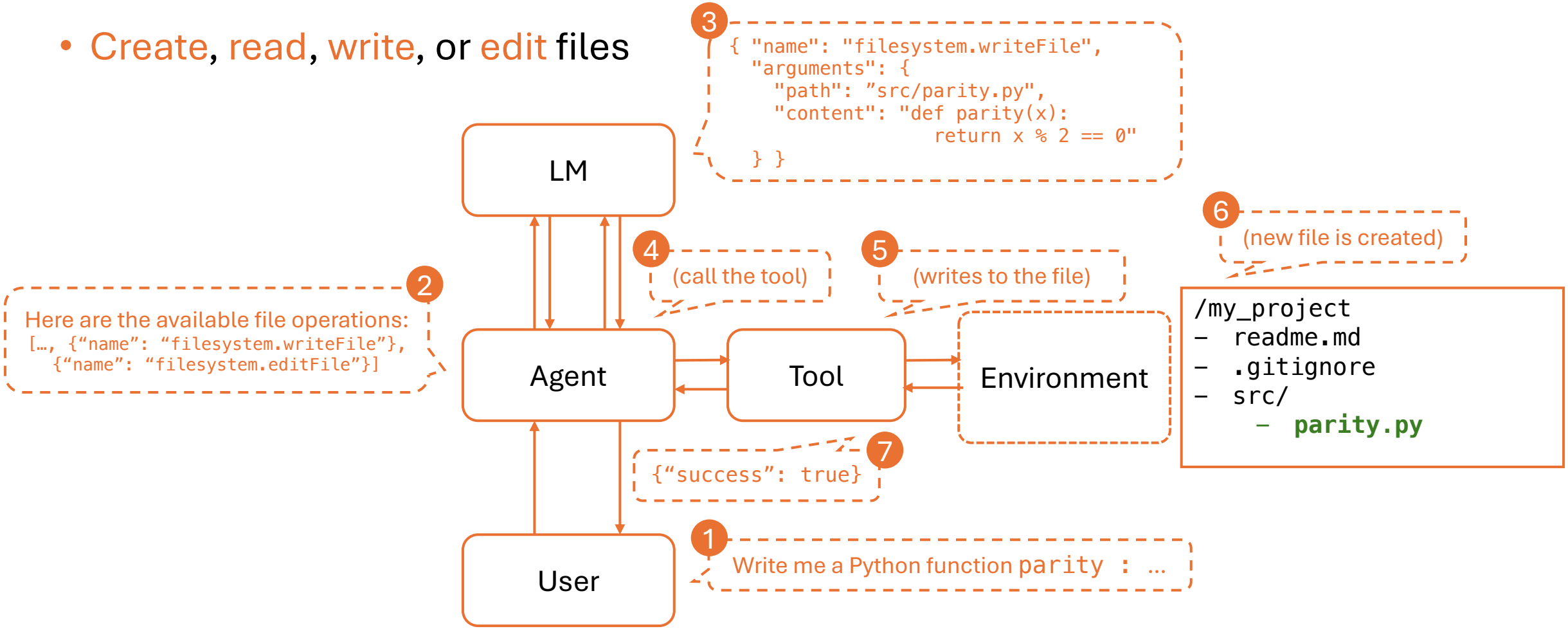
File System (FS) for Code Implementation

- Create, read, write, or edit files



File System (FS) for Code Implementation

- Create, read, write, or edit files



```
{
  name: "read_text_file",
  description:
    "Read the complete contents of a file from the file system as text. " +
    "Handles various text encodings and provides detailed error messages " +
    "if the file cannot be read. Use this tool when you need to examine " +
    "the contents of a single file.
    "the first N lines of a file, o
    "the last N lines of a file. Op
    "Only works within allowed dire
  inputSchema: zodToJsonSchema(Read
},
```

```
{
  name: "write_file",
  description:
    "Create a new file or completely overwrite an existing file with new content. " +
    "Use with caution as it will overwrite existing files without warning. " +
    "Handles text content with proper encoding. Only works within allowed directories.",
```

```
{
  name: "create_directory",
  description:
    "Create a new directory or ensure a directory exists. Can create multiple " +
    "nested directories in one operation. If the directory already exists, " +
    "this operation will succeed silently. Perfect for setting up directory " +
    "structures for projects or ensuring required paths exist. Only works within allowed directories.",
  inputSchema: zo
},
```

```
{
  name: "directory_tree",
  description:
      "Get a recursive tree view of files and directories as a JSON structure. " +
      "Each entry includes 'name', 'type' (file/directory), and 'children' for directories. " +
      "Files have no children array, while directories always have a children array (which may be empty). " +
      "The output is formatted with 2-space indentation for readability. Only works within allowed directories.",
  inputSchema: zodToJsonSchema(DirectoryTreeArgsSchema) as ToolInput,
},
```

https://github.com/modelcontextprotocol/servers/blob/main/src/filesystem/index.ts

File System (FS) for Code Implementation

- Edit file: how do we specify the “edit” action?
  - Choice 1: Rewrite the entire file
    - Will waste a
    - Edit: new file content S
  - Choice 2: Rewrite given lines by line numbers
    - Need to also display “line numbers” when reading the file
    - Edit: change lines X to Y by the string S
  - Choice 3: Replace a string with another string
    - Is a bit fragile; the pattern string must be chosen carefully:
      - If it’s too generic, the tool may unintentionally modify multiple occurrences.
      - If there is no exact match, nothing can be edited.
    - Edit: replace string X with string Y

File System (FS) for Code Implementation

- Edit file: how do we specify the “edit” action?
  - Choice 1: Rewrite the entire file
    - Will waste a
    - Edit: new file content S
  - Choice 2: Rewrite given lines by line numbers
    - Need to also display “line numbers” when reading the file
    - Edit: change lines X to Y by the string S
  - Choice 3: Replace a string with another string
    - Is a bit fragile; the pattern string must be chosen carefully:
      - If it’s too generic, the tool may unintentionally modify multiple occurrences.
      - If there is no exact match, nothing can be edited.
    - Edit: replace string X with string Y

SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering

John Yang* **Carlos E. Jimenez*** **Alexander Wettig** **Kilian Lieret**
Shunyu Yao **Karthik Narasimhan** **Ofir Press**

Princeton Language and Intelligence, Princeton University

- Choice 2: Rewrite given lines by line numbers
  - Need to also display “line numbers” when reading the file
  - Edit: change lines X to Y by the string S
- Choice 3: Replace a string with another string
  - Is a bit fragile; the pattern string must be chosen carefully:
    - If it’s too generic, the tool may unintentionally modify multiple occurrences.
    - If there is no exact match, nothing can be edited.
  - Edit: replace string X with string Y

SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering, Yang et. al., NeurIPS 2024

SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering

John Yang* Carlos E. Jimenez* Alexander Wettig Kilian Lieret
Shunyu Yao Karthik Narasimhan Ofir Press
Princeton Language and Intelligence, Princeton University

Observation (File Viewer)

```
[File: solvers/diophantine.py (3093 lines total)]
(400 lines above)
401:    elif total_degree == 3:
402:        diop_type = "cubic_thue"
403:
404:    elif (total_degree > 3):
405:        if all(coeff[k] == 1):
406:            diop_type = 'general_sum_even'
407:
408:    if diop_type is not None:
409:        return var, coeff, diop_type
(2684 lines below)
```

(a) Observation from the file viewer.

Response (Thought + Edit Action)

We need to add a check to ensure that all powers are the same. Let's edit the code to include this check.

```
edit 404:407
    elif (total_degree > 3 and
          len(set(k.exp[k.is_Pow])) == 1):
        if all(coeff[k] == 1):
            diop_type = 'general_sum_even'
end_of_edit
```

(b) Action using the edit interface.

Figure 3: The file viewer and edit command are integrated. (a) The file viewer shows the agent the open file's content with line numbers. (b) The agent invokes the edit function to replace lines 404-407 in the open file. After the edit, the file viewer shows the agent the now updated version of the file.

- Edit file: how
  - Choice 1: Re
    - Will waste
    - Edit: new f
- Choice 2: Rewrite given lines by line numbers
  - Need to also display "line numbers" when reading the file
  - Edit: change lines X to Y by the string S
- Choice 3: Replace a string with another string
  - Is a bit fragile; the pattern string must be chosen carefully:
    - If it's too generic, the tool may unintentionally modify multiple occurrences.
    - If there is no exact match, nothing can be edited.
  - Edit: replace string X with string Y

SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering, Yang et. al., NeurIPS 2024

File System (FS) for Code Implementation

- Edit file: how do we specify the "edit" action?
  - Choice 1: Rewrite the entire file
    - Will waste a
    - Edit: new file content S
  - Choice 2: Rewrite given lines by line numbers
    - Need to also display "line numbers" when reading the file
    - Edit: change lines X to Y by the string S
  - Choice 3: Replace a string with another string
    - Is a bit fragile; the pattern string must be chosen carefully:
      - If it's too generic, the tool may unintentionally modify multiple occurrences.
      - If there is no exact match, nothing can be edited.
    - Edit: replace string X with string Y

File System (FS) for Code Implementation

- Edit file: how do we specify the "edit" action?
  - Choice
    - Will
    - Edit:
  - Choice
    - Need
    - Edit:

```
102    const EditOperation = z.object({
103      oldText: z.string().describe('Text to search for - must match exactly'),
104      newText: z.string().describe('Text to replace with')
105    });
106
107    const EditFileArgsSchema = z.object({
108      path: z.string(),
109      edits: z.array(EditOperation),
110      dryRun: z.boolean().default(false).describe('Preview changes using git-style diff format')
111    });
```

  - Choice 3: Replace a string with another string
    - Is a bit fragile; the pattern string must be chosen carefully:
      - If it's too generic, the tool may unintentionally modify multiple occurrences.
      - If there is no exact match, nothing can be edited.
    - Edit: replace string X with string Y

https://github.com/modelcontextprotocol/servers/blob/main/src/filesystem/index.ts

SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering

John Yang* **Carlos E. Jimenez*** **Alexander Wettig** **Kilian Lieret**
Shunyu Yao **Karthik Narasimhan** **Ofir Press**

Princeton Language and Intelligence, Princeton University

SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering, Yang et. al., NeurIPS 2024

SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering

John Yang* Ca
Shuny
Princeton

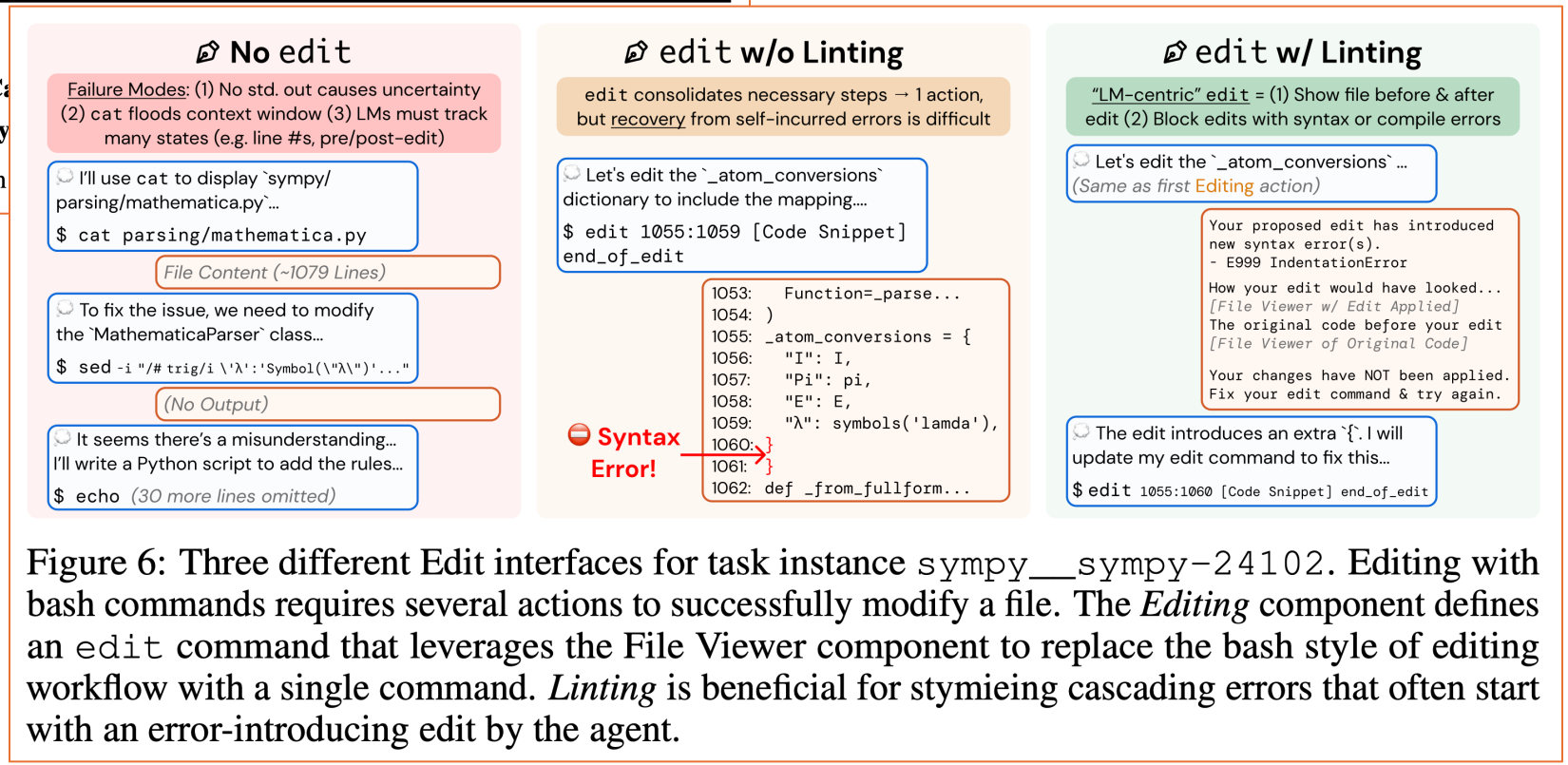


Figure 6: Three different Edit interfaces for task instance `sympy__sympy-24102`. Editing with bash commands requires several actions to successfully modify a file. The *Editing* component defines an `edit` command that leverages the File Viewer component to replace the bash style of editing workflow with a single command. *Linting* is beneficial for stymieing cascading errors that often start with an error-introducing edit by the agent.

SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering, Yang et. al., NeurIPS 2024

SWE-agent: Agent-Computer Interfaces Enable

Figure 6: Three bash commands an edit comm workflow with a with an error-in

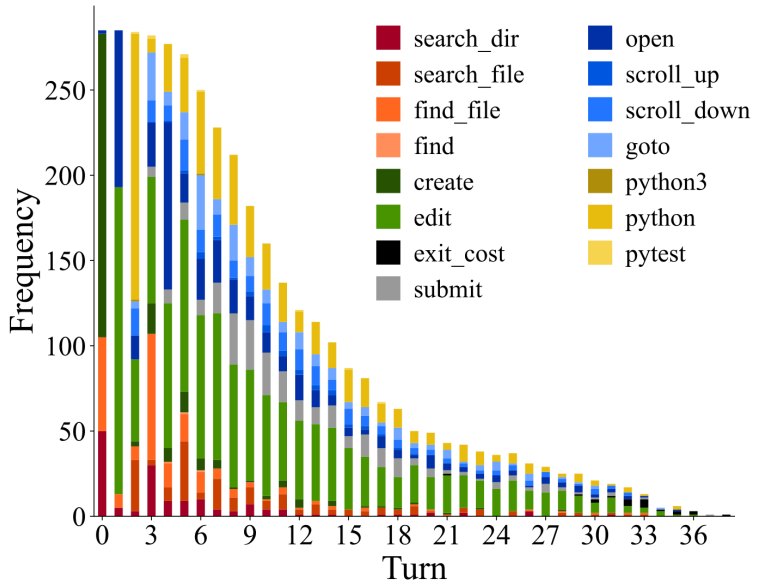


Figure 7: The frequency with which actions are invoked at each turn by SWE-agent w/ GPT-4 for task instances that it solved on the SWE-bench full test set (286 trajectories).

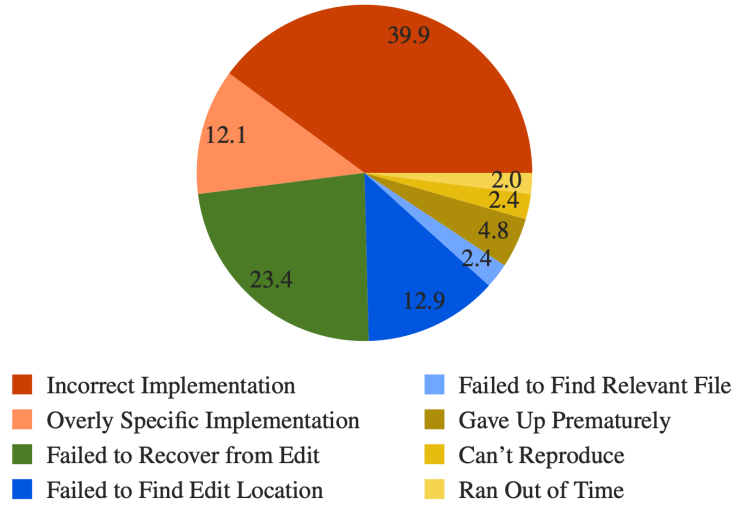


Figure 8: Failure mode distribution for SWE-agent w/ GPT-4 Turbo trajectories of unresolved instances. Each instance is labeled automatically using an LM with the categories from Table 9.

SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering, Yang et. al., NeurIPS 2024

Can you look at core/scallop-fir/docs/ and help me implement a type_checker

I'll help you implement a type checker for the Scallop language. Let me first explore the documentation and understand the current structure of the project.

Listed docs

Read grammar.md

Searched What is the current

Let me explore the existing

Read test_type_inference.rs

Read test_type_bounds.rs

Searched Where are type def

Let me explore the source c

Listed src

Read lib.rs

Grepped type.*check|TypeChecker|type_infere

I can see that the type checking functionalit

s . Let me explore the AST types to understa

Let me explore more of the type system structure:

Read mod.rs

Read type_decl.rs

Read named_type.rs

Now let me create a comprehensive plan for implementing the type checker:

1 of 9 Done

~~Explore AST structure and understand type system components~~

Create the analysis module with type checking infrastructure

Implemer

Now let me create the analysis module with the type checking infrastructure:

mod.rs (new) +3 •

type_inference.rs (new) +510 •

Now let me update the lib.rs to enable t

lib.rs +1 -1 •

Let me check for any compilation errors

No linter errors found in analysis

I need to fix the generic type parameter structure. Let me check the actual structure:

type_inference.rs +1 -1 •

Now let me check the ADT declaration structure:

Read adt_decl.rs

I need to fix the ADT checking code:

type_inference.rs +7 -2 •

Let me check the alias type declaration structure:

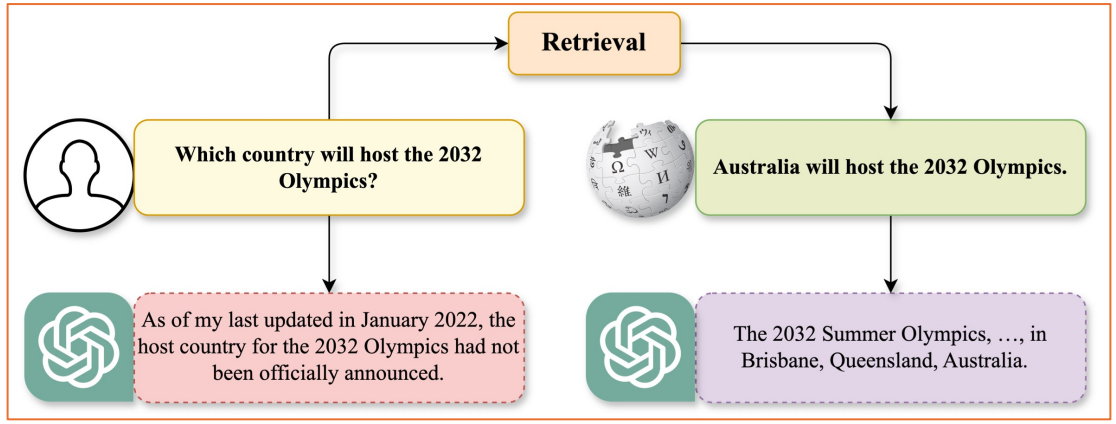
Read alias_type_decl.rs

I need to fix the alias type declaration checking:

type_inference.rs +7 -1 •

Retrieval Augmented Generation (RAG)

- Intuition:
  - We do not solely rely on Language Model's inert knowledge
  - We assume that there is an external database of knowledge
    - Each piece of knowledge is called a "document"
  - Given a task, we retrieve some documents from this database
  - Based on the documents, we ask LM to continue generating the answers



The Survey of Retrieval-Augmented Text Generation in Large Language Models, Huang et. al., 2024

Retrieval Augmented Generation (RAG)

Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Patrick Lewis[†‡], **Ethan Perez**[⋆],

Aleksandra Piktus[†], **Fabio Petroni**[†], **Vladimir Karpukhin**[†], **Naman Goyal**[†], **Heinrich Küttler**[†],

Mike Lewis[†], **Wen-tau Yih**[†], **Tim Rocktäschel**[†‡], **Sebastian Riedel**[†‡], **Douwe Kiela**[†]

[†]Facebook AI Research; [‡]University College London; [⋆]New York University;
plewis@fb.com

Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, Lewis et. al., NeurIPS 2020

Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

P

Aleksandra Piktus†, Fabio Petr

Mike Lewis†, Wen-tau Yih†

†Facebook AI Research

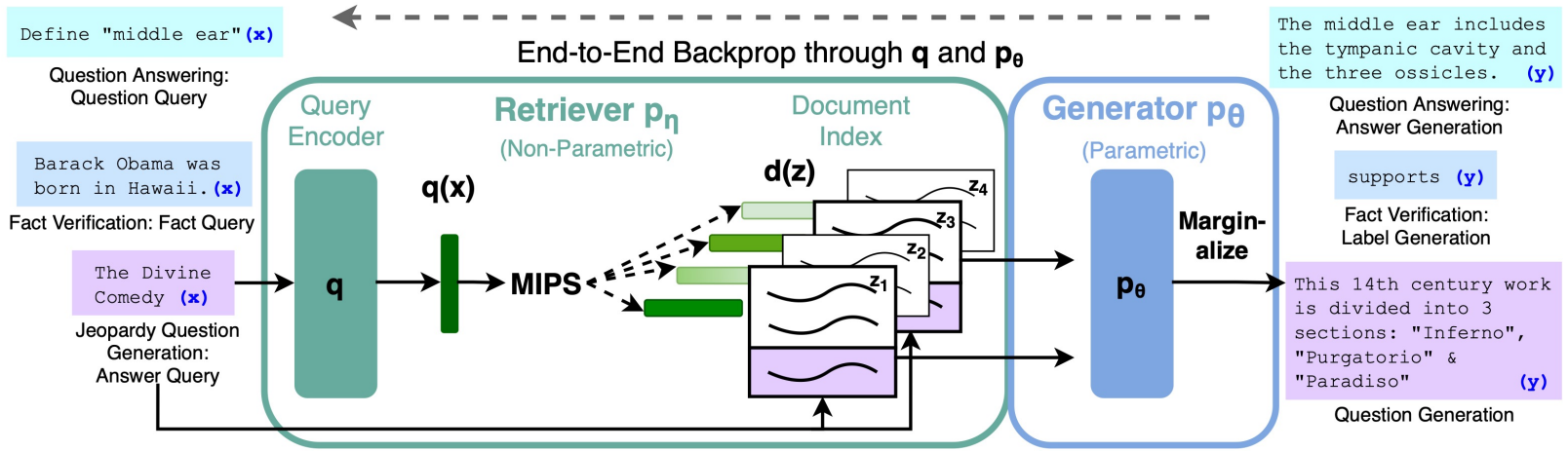


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat z as a latent variable and marginalize over seq2seq predictions given different documents.

Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, Lewis et. al., NeurIPS 2020

Retrieval-Augmented Generation for

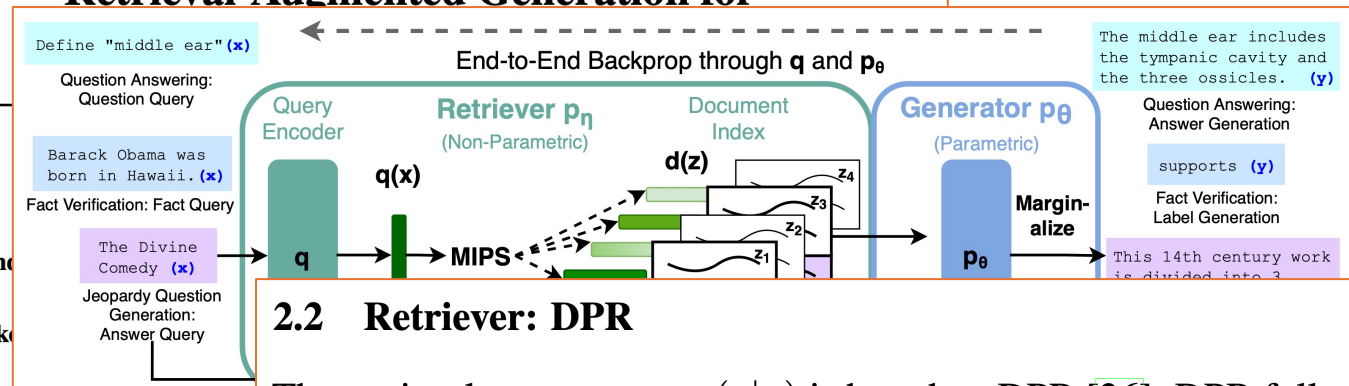


Figure 1: Overvie
Index) with a pre-
Maximum Inner I
treat z as a latent

2.2 Retriever: DPR

The retrieval component $p_\eta(z|x)$ is based on DPR [26]. DPR follows a bi-encoder architecture:

$$p_\eta(z|x) \propto \exp\left(\mathbf{d}(z)^\top \mathbf{q}(x)\right) \qquad \mathbf{d}(z) = \text{BERT}_d(z), \quad \mathbf{q}(x) = \text{BERT}_q(x)$$

where $\mathbf{d}(z)$ is a dense representation of a document produced by a $\text{BERT}_{\text{BASE}}$ *document encoder* [8], and $\mathbf{q}(x)$ a query representation produced by a *query encoder*, also based on $\text{BERT}_{\text{BASE}}$. Calculating top-k$(p_\eta(\cdot|x))$, the list of k documents z with highest prior probability $p_\eta(z|x)$, is a Maximum Inner Product Search (MIPS) problem, which can be approximately solved in sub-linear time [23]. We use a pre-trained bi-encoder from DPR to initialize our retriever and to build the document index. This retriever was trained to retrieve documents which contain answers to TriviaQA [24] questions and Natural Questions [29]. We refer to the document index as the *non-parametric memory*.

The Survey of Retrieval-Augmented Text Generation in Large Language Models

YIZHENG HUANG and JIMMY X. HUANG, York University, Canada

Retrieval-Augmented Generation (RAG) merges retrieval methods with deep learning advancements to address the static limitations of large language models (LLMs) by enabling the dynamic integration of up-to-date external information. This methodology, focusing primarily on the text domain, provides a cost-effective solution to the generation of plausible but possibly incorrect responses by LLMs, thereby enhancing the accuracy and reliability of their outputs through the use of real-world data. As RAG grows in complexity and incorporates multiple concepts that can influence its performance, this paper organizes the RAG paradigm into four categories: pre-retrieval, retrieval, post-retrieval, and generation, offering a detailed perspective from the retrieval viewpoint. It outlines RAG's mechanics and discusses the field's progression through the analysis of significant studies. Additionally, the paper introduces evaluation methods for RAG, addressing the challenges faced and proposing future research directions. By offering an organized framework and categorization, the study aims to consolidate existing research on RAG, clarify its technological underpinnings, and highlight its potential to broaden the adaptability and applications of LLMs.

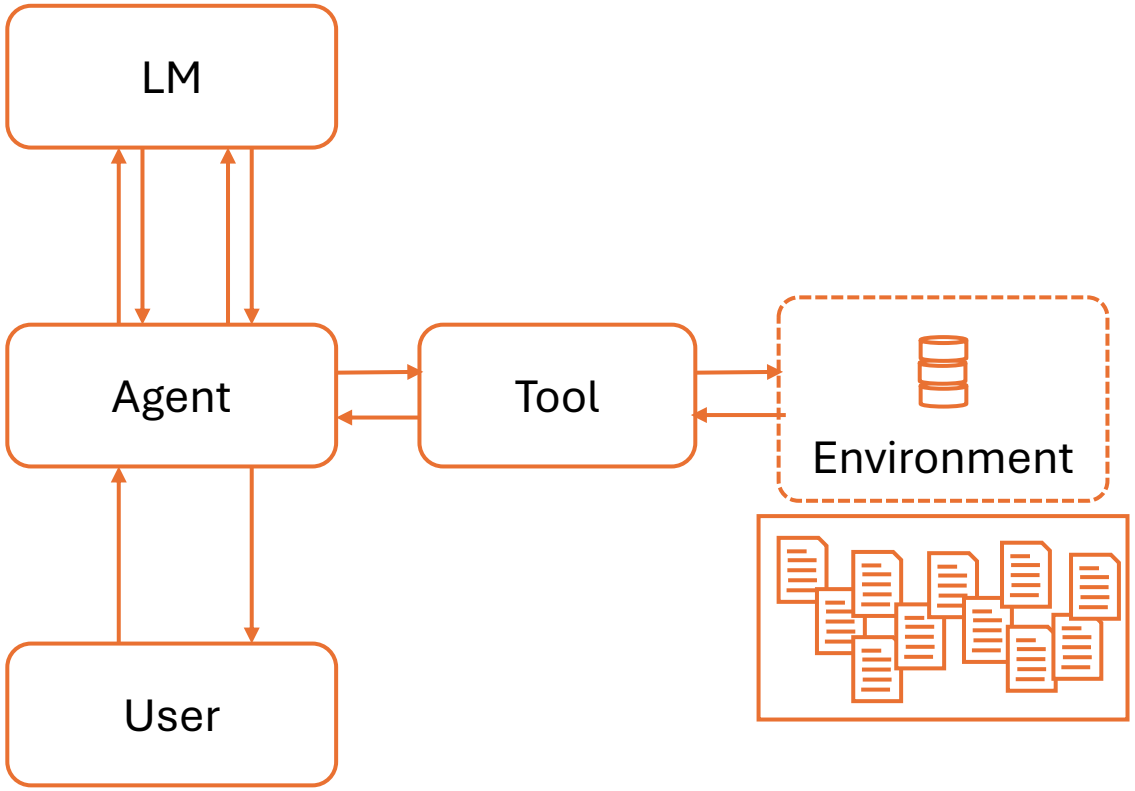
The Survey of Retrieval-Augmented Text Generation in Large Language Models, Huang et. al., 2024

Retrieval Augmented Generation for Code

- Vector database as environment:
  - Each document z in the database is associated with an embedding $d(z)$
  - Querying: embedding-based similarity search
    - Assume query x, query embedding is $q(x)$
    - Similarity score is cosine similarity between $q(x)$ and $d(z)$:

$$\text{sim}(x, z) = \frac{q(x) \cdot d(z)}{\|q(x)\| \cdot \|d(z)\|}$$

  - We can add, update, or remove documents using document metadata

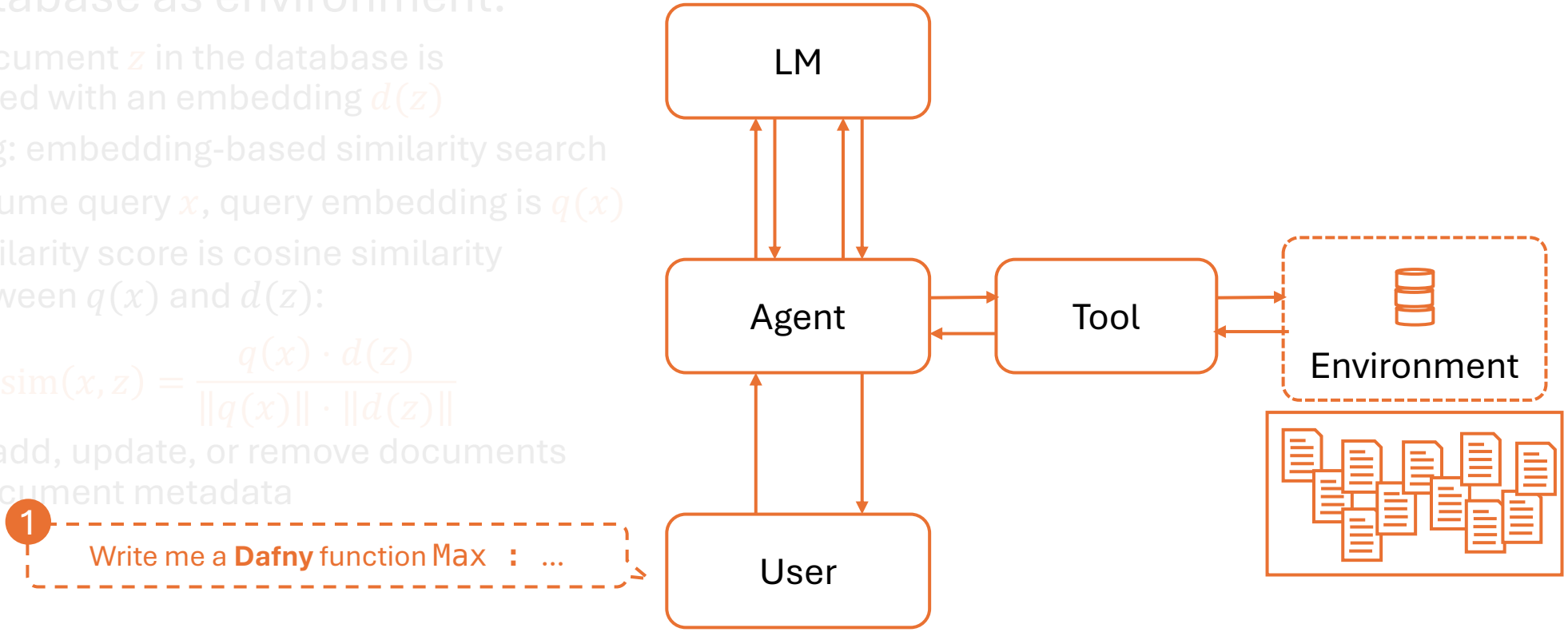


Retrieval Augmented Generation for Code

- Vector database as environment:
  - Each document z in the database is associated with an embedding $d(z)$
  - Querying: embedding-based similarity search
    - Assume query x, query embedding is $q(x)$
    - Similarity score is cosine similarity between $q(x)$ and $d(z)$:

$$\text{sim}(x, z) = \frac{q(x) \cdot d(z)}{\|q(x)\| \cdot \|d(z)\|}$$

  - We can add, update, or remove documents using document metadata

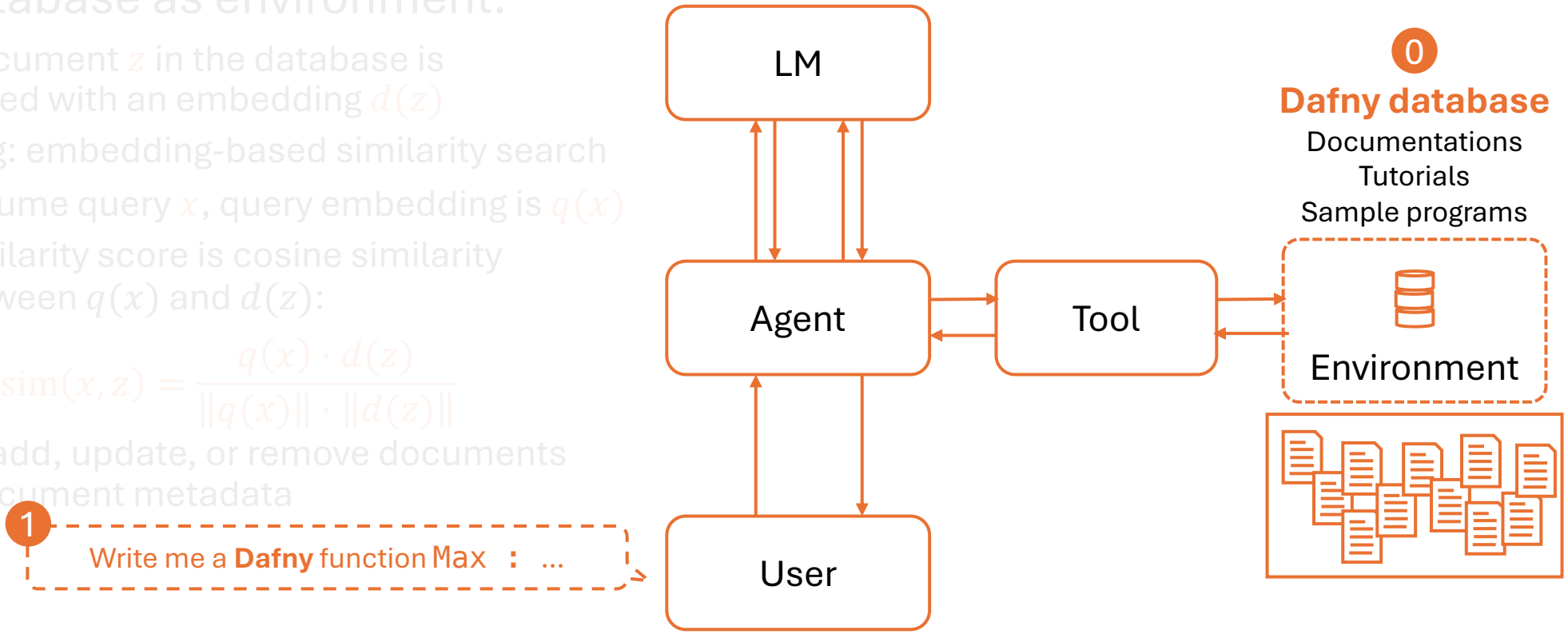


Retrieval Augmented Generation for Code

- Vector database as environment:
  - Each document z in the database is associated with an embedding $d(z)$
  - Querying: embedding-based similarity search
    - Assume query x, query embedding is $q(x)$
    - Similarity score is cosine similarity between $q(x)$ and $d(z)$:

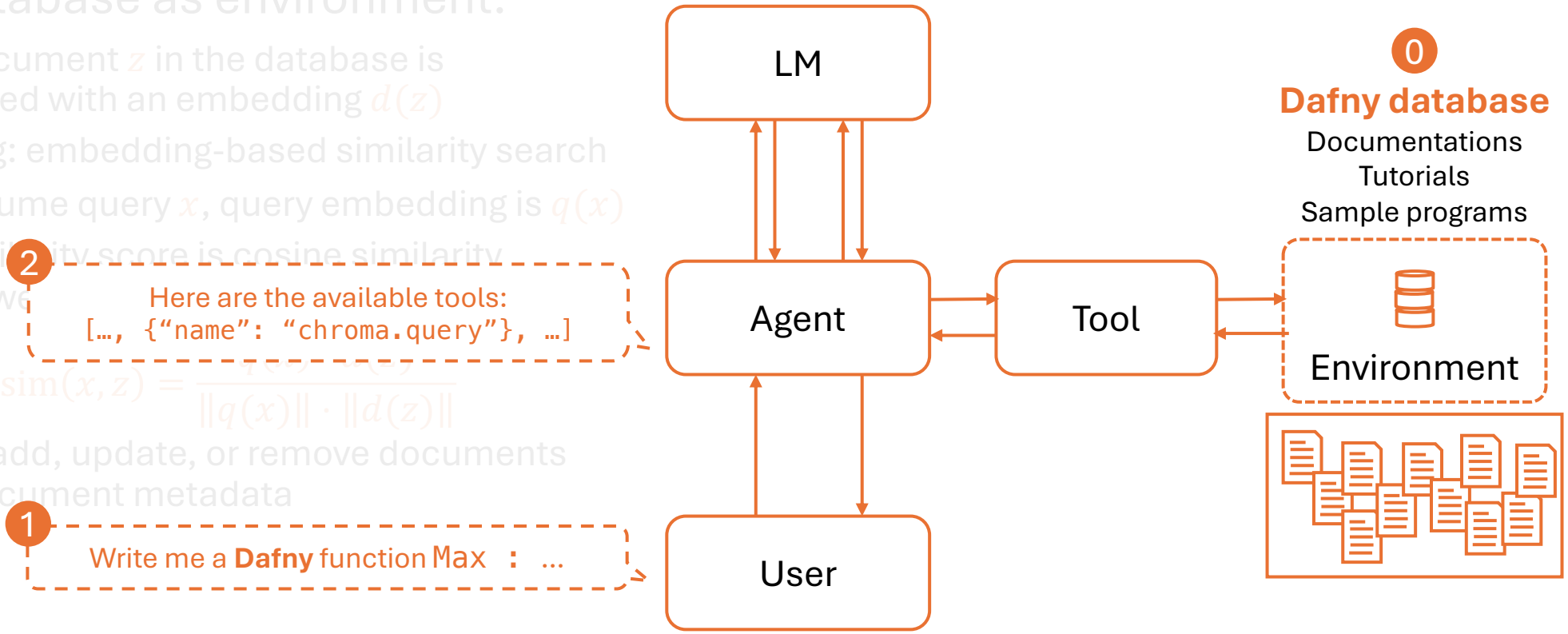
$$\text{sim}(x, z) = \frac{q(x) \cdot d(z)}{\|q(x)\| \cdot \|d(z)\|}$$

  - We can add, update, or remove documents using document metadata

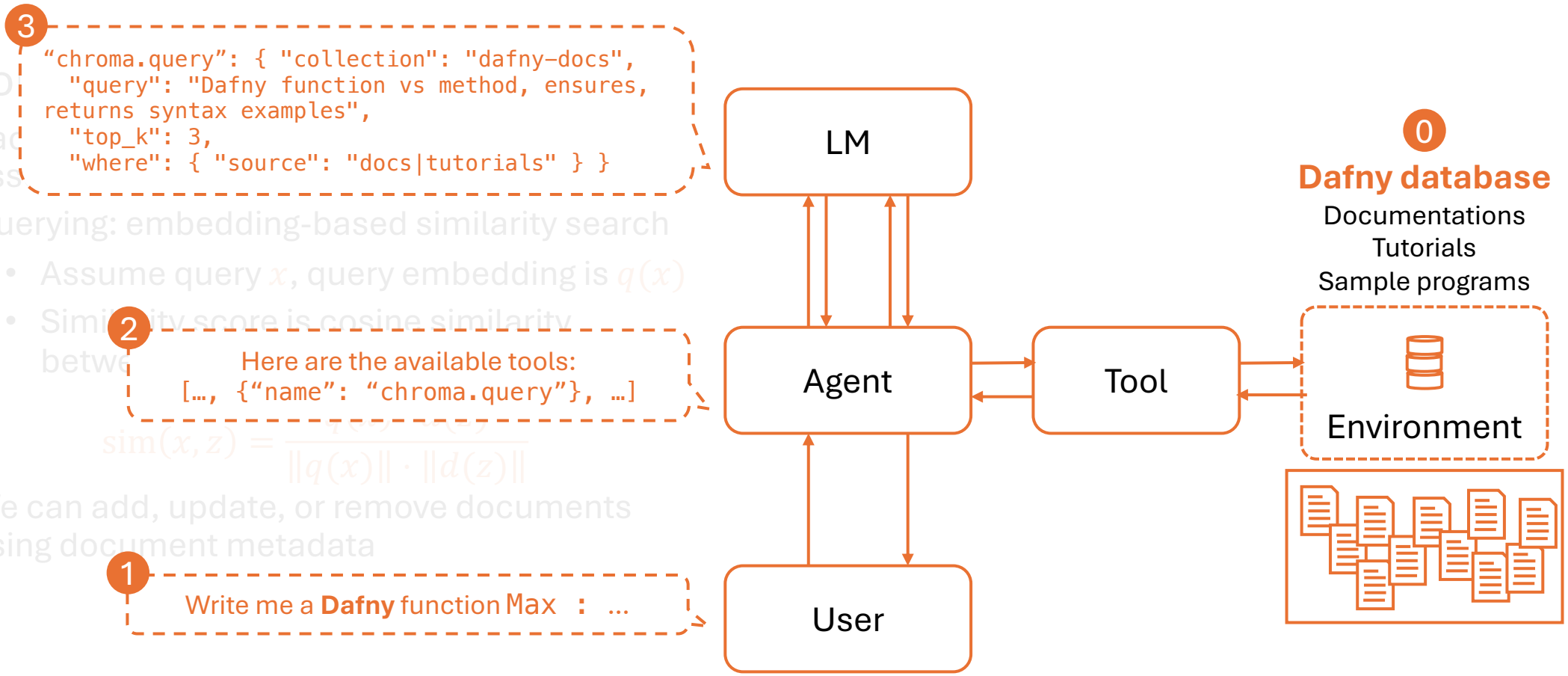


Retrieval Augmented Generation for Code

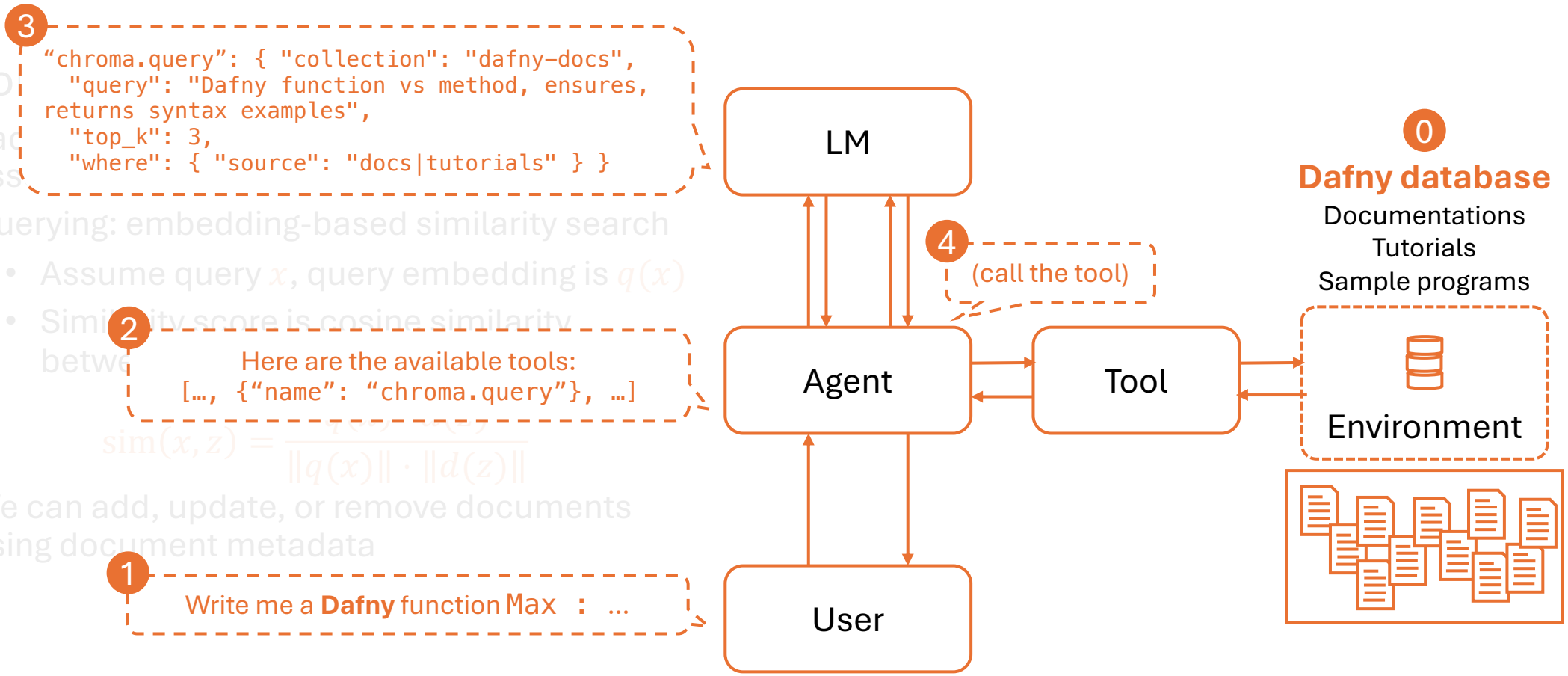
- Vector database as environment:
  - Each document z in the database is associated with an embedding $d(z)$
  - Querying: embedding-based similarity search
    - Assume query x, query embedding is $q(x)$
    - Similarity score is cosine similarity between $q(x)$ and $d(z)$: $\text{sim}(x, z) = \frac{q(x) \cdot d(z)}{\|q(x)\| \cdot \|d(z)\|}$
  - We can add, update, or remove documents using document metadata



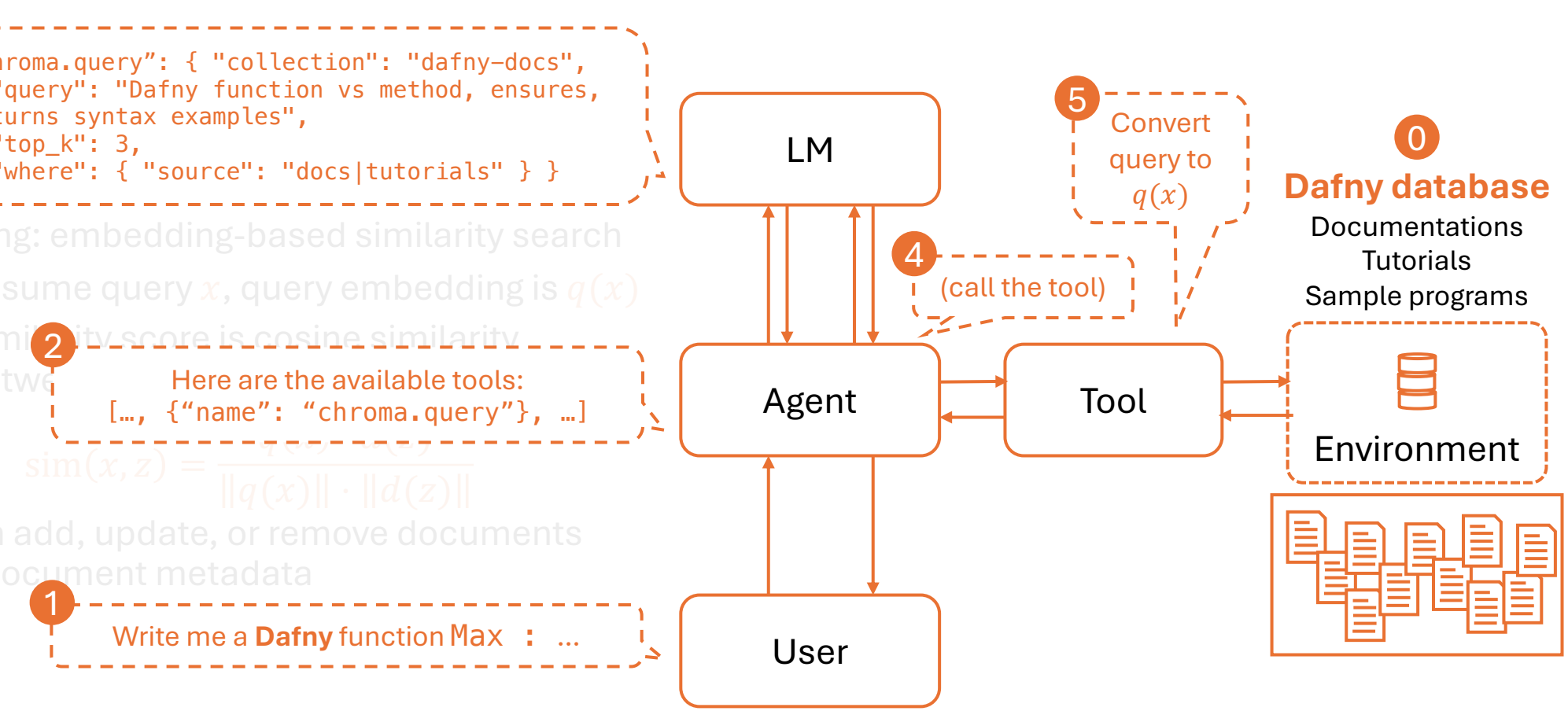
Retrieval Augmented Generation for Code



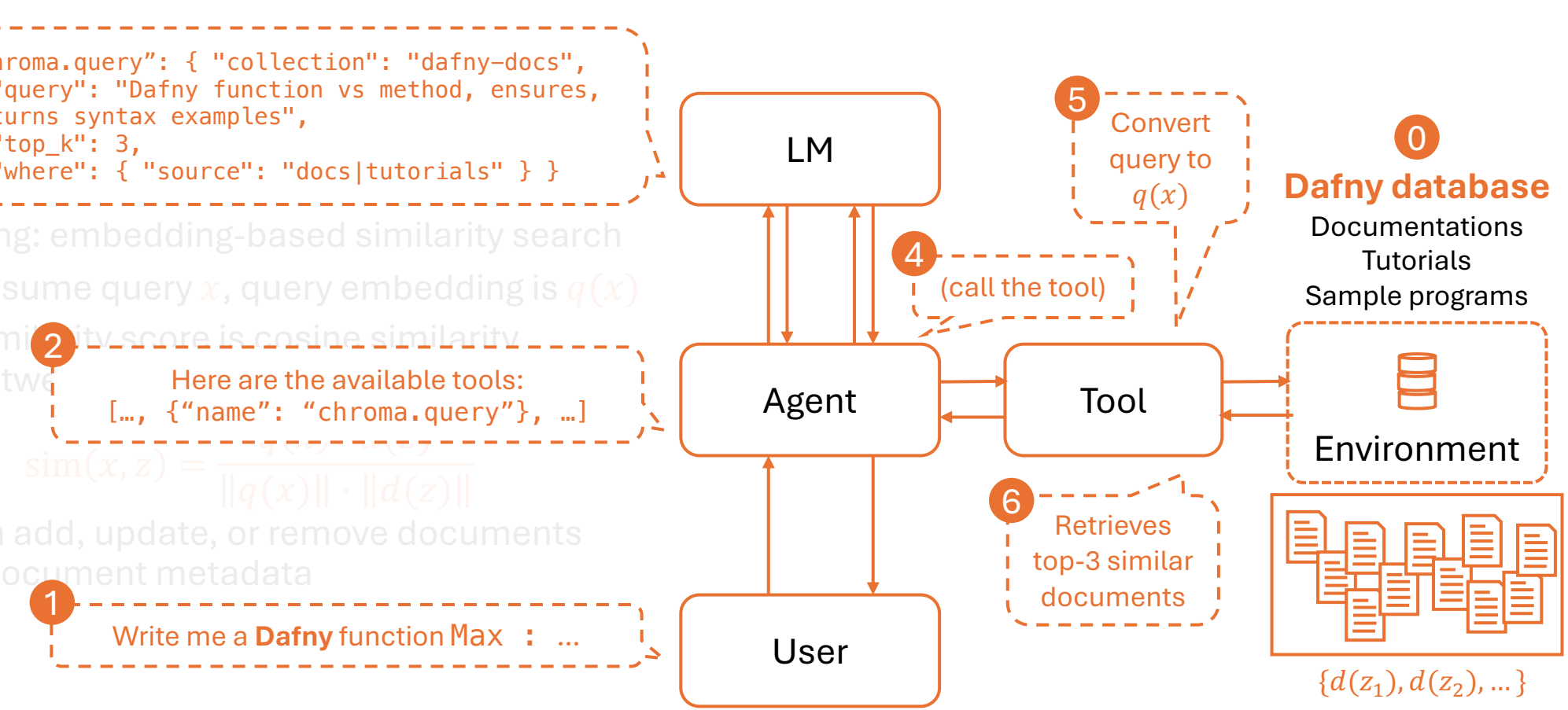
Retrieval Augmented Generation for Code



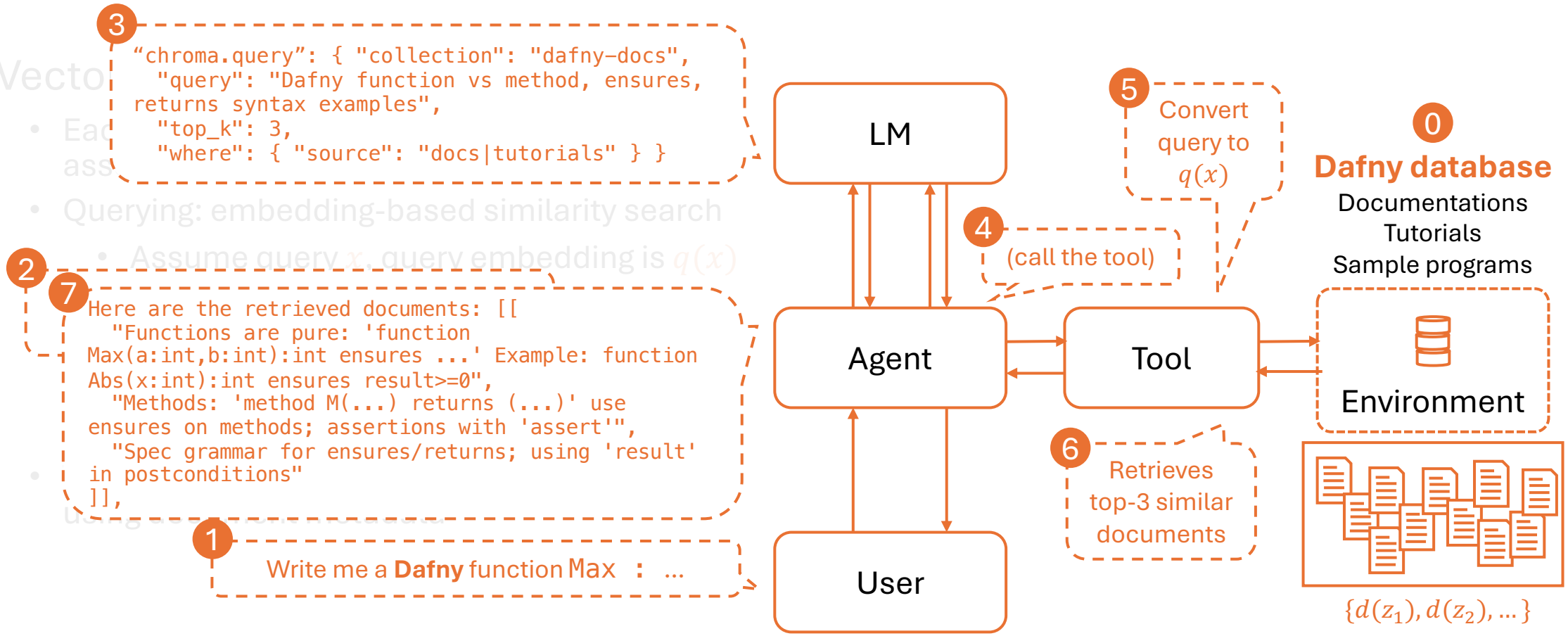
Retrieval Augmented Generation for Code



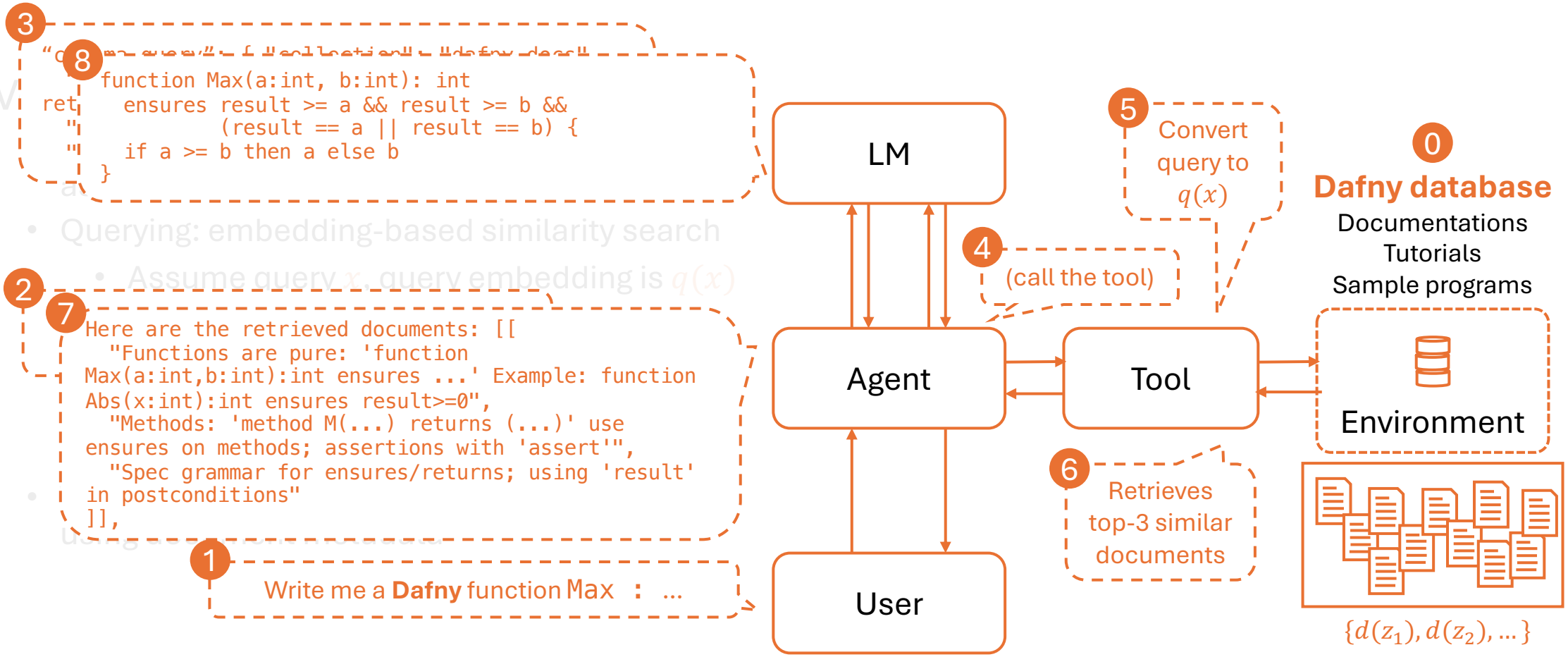
Retrieval Augmented Generation for Code



Retrieval Augmented Generation for Code



Retrieval Augmented Generation for Code

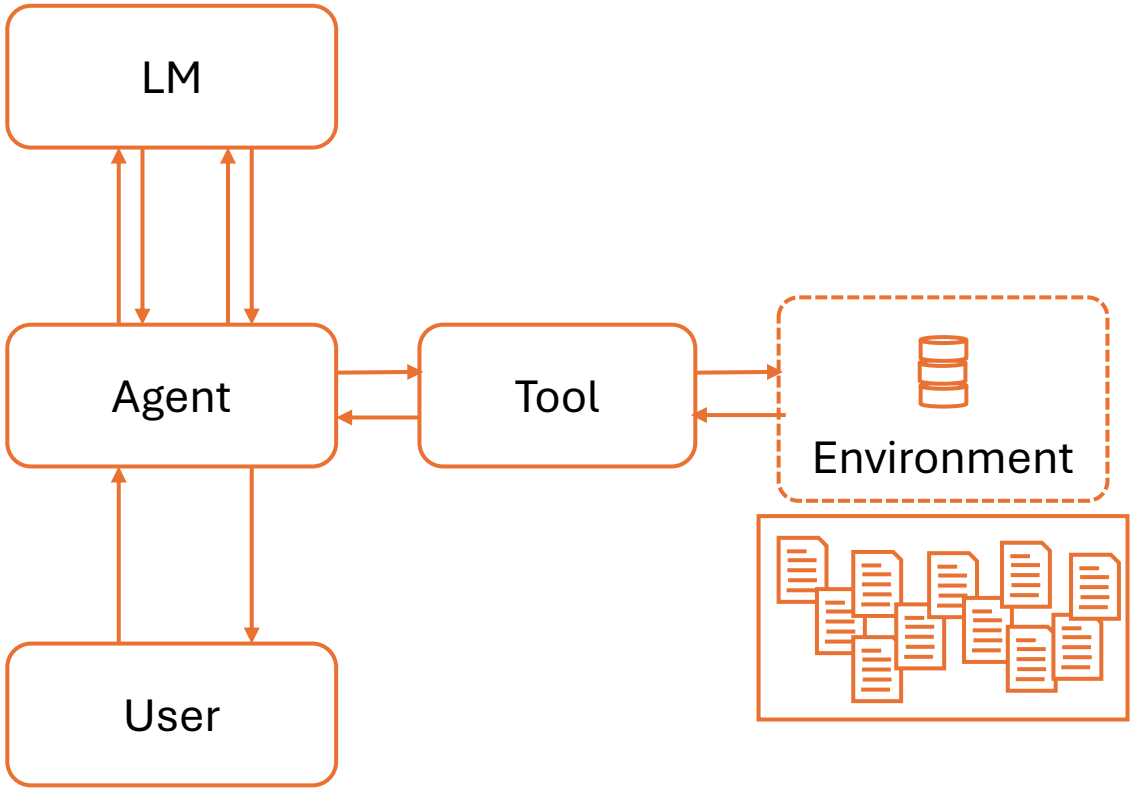


Retrieval Augmented Generation for Code

- Vector database as environment:
  - Each document z in the database is associated with an embedding $d(z)$
  - Querying: embedding-based similarity search
    - Assume query x, query embedding is $q(x)$
    - Similarity score is cosine similarity between $q(x)$ and $d(z)$:

$$\text{sim}(x, z) = \frac{q(x) \cdot d(z)}{\|q(x)\| \cdot \|d(z)\|}$$

  - We can add, update, or remove documents using document metadata



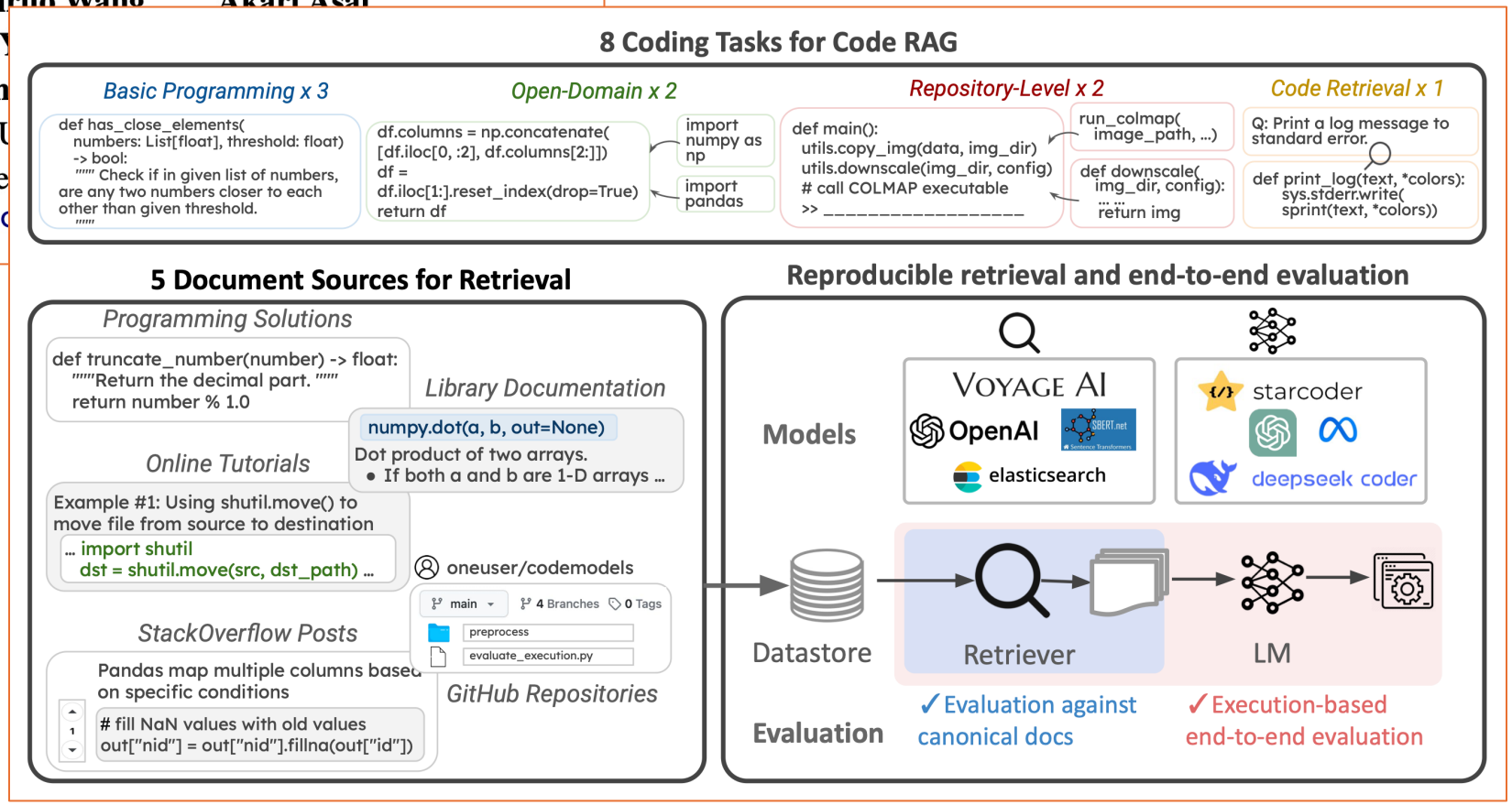
CODERAG-BENCH: Can Retrieval Augment Code Generation?

Zora Zhiruo Wang[♠*] **Akari Asai**[◇*]
Xinyan Velocity Yu[♡] **Frank F. Xu**[♠] **Yiqing Xie**[♠]
Graham Neubig[♠] **Daniel Fried**[♠]
[♠]Carnegie Mellon University [◇]University of Washington
[♡]University of Southern California
https://code-rag-bench.github.io/

CodeRAG-Bench: Can Retrieval Augment Code Generation? Wang et. al., NAACL 2025 Findings

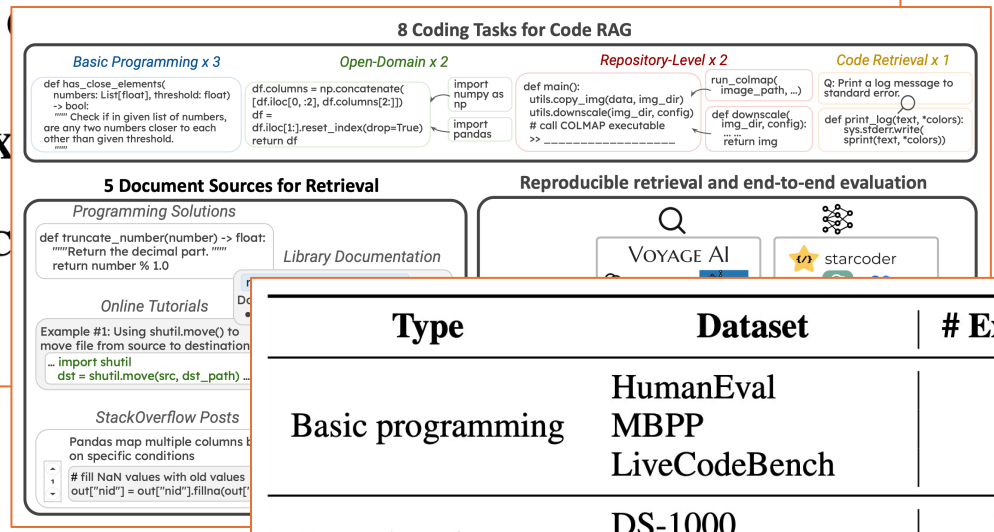
CODERAG-BENCH: Can Retrieval Augment Code Generation?

Zora Zhiruo Wang[♠*] Akari Asai[♢*]
Xinyan Velocity Y
Graham
[♠]Carnegie Mellon U
[♡]Unive
https://c



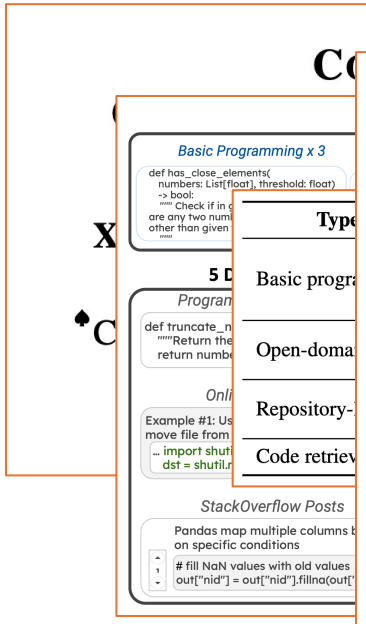
CodeRAG-Bench: Can Retrieval Augment Code Generation? Wang et. al., NAACL 2025 Findings

CODERAG-BENCH:



Type	Dataset	# Examples	# Corpus	Ground-Truth Docs	Evaluation
Basic programming	HumanEval	164	164	program solutions	execution
	MBPP	500	500	program solutions	execution
	LiveCodeBench	400	-	-	execution
Open-domain	DS-1000	1000	34,003	docs	execution
	ODEX	945	34,003	docs, stackoverflow	execution
Repository-level	RepoEval (function)	373	237	github repository	execution
	SWE-bench-Lite	300	40,868	github repository	execution
Code retrieval	CodeSearchNet-Py	22,177	22177	CSN functions	ndcg@10

CodeRAG-Bench: Can Retrieval Augment Code Generation? Wang et. al., NAACL 2025 Findings



Method	Basic Programming			Open-Domain			Repo-Level	
	HumanEval	MBPP	LCB	DS-1000	ODEX	ODEX-hard	RepoEval	SWE-bench
w/ StarCoder2-7B								
None	31.7	2.4	1.5	29.2	14.6	10.3	26.5	0.0
BM25	**43.9**	51.8	1.0	**36.7**	14.1	13.8	36.7	0.0
GIST-large	38.7	50.4	0.5	35.9	**17.3**	13.8	40.8	0.3
Voyage, code	39.0	52.6	0.3	36.0	15.3	10.3	45.8	0.3
OpenAI, small	39.0	52.6	**1.5**	35.5	15.9	**17.2**	51.2	0.0
OpenAI, rerank	34.8	**53.4**	0.5	33.4	14.1	**17.2**	**53.9**	0.3
Gold	94.5	34.8	-	30.0	17.5	17.2	42.0	0.7
w/ DeepseekCoder-7B-instruct								
None	70.1	60.8	30.5	41.4	39.2	17.2	28.2	0.7
BM25	**68.9**	60.0	31.8	**36.6**	37.8	20.7	37.3	0.0
GIST-large	66.3	56.6	**33.8**	35.9	34.9	20.7	44.5	0.3
Voyage, code	66.5	56.4	31.8	35.9	**39.4**	17.2	46.6	0.3
OpenAI, small	**68.9**	58.6	32.0	35.5	37.1	20.7	55.2	0.3
OpenAI, rerank	53.0	**60.6**	31.5	36.5	37.1	**24.1**	**55.5**	0.3
Gold	87.8	63.6	-	43.2	41.7	24.1	48.1	0.0
w/ GPT-3.5-turbo								*GPT-4o*
None	72.6	70.8	35.3	43.7	41.7	17.2	23.9	2.3
BM25	73.2	72.4	**35.5**	36.9	**41.0**	**24.1**	30.8	6.7
GIST-large	73.2	68.2	34.8	36.7	36.2	13.8	38.3	19.3
Voyage, code	**75.0**	66.8	34.5	**37.4**	**41.0**	20.7	43.2	15.7
OpenAI, small	73.8	68.4	35.8	36.9	40.3	17.2	48.0	21.0
OpenAI, rerank	64.0	**72.6**	33.5	**37.4**	40.5	17.2	**49.6**	**21.7**
Gold	91.5	72.6	-	42.9	40.3	24.1	39.1	30.7

Table 6: Performance of retrieval-augmented code generation, with top retrieval and generation models. We bold-type the best RACG results. We test gpt-4o on SWE-bench to show non-trivial results than gpt-3.5-turbo. Note that we exclude code canonical answer from the retrieval corpora for basic programming tasks.

CodeRAG-Bench: Can Retrieval Augment Code Generation? Wang et. al., NAACL 2025 Findings

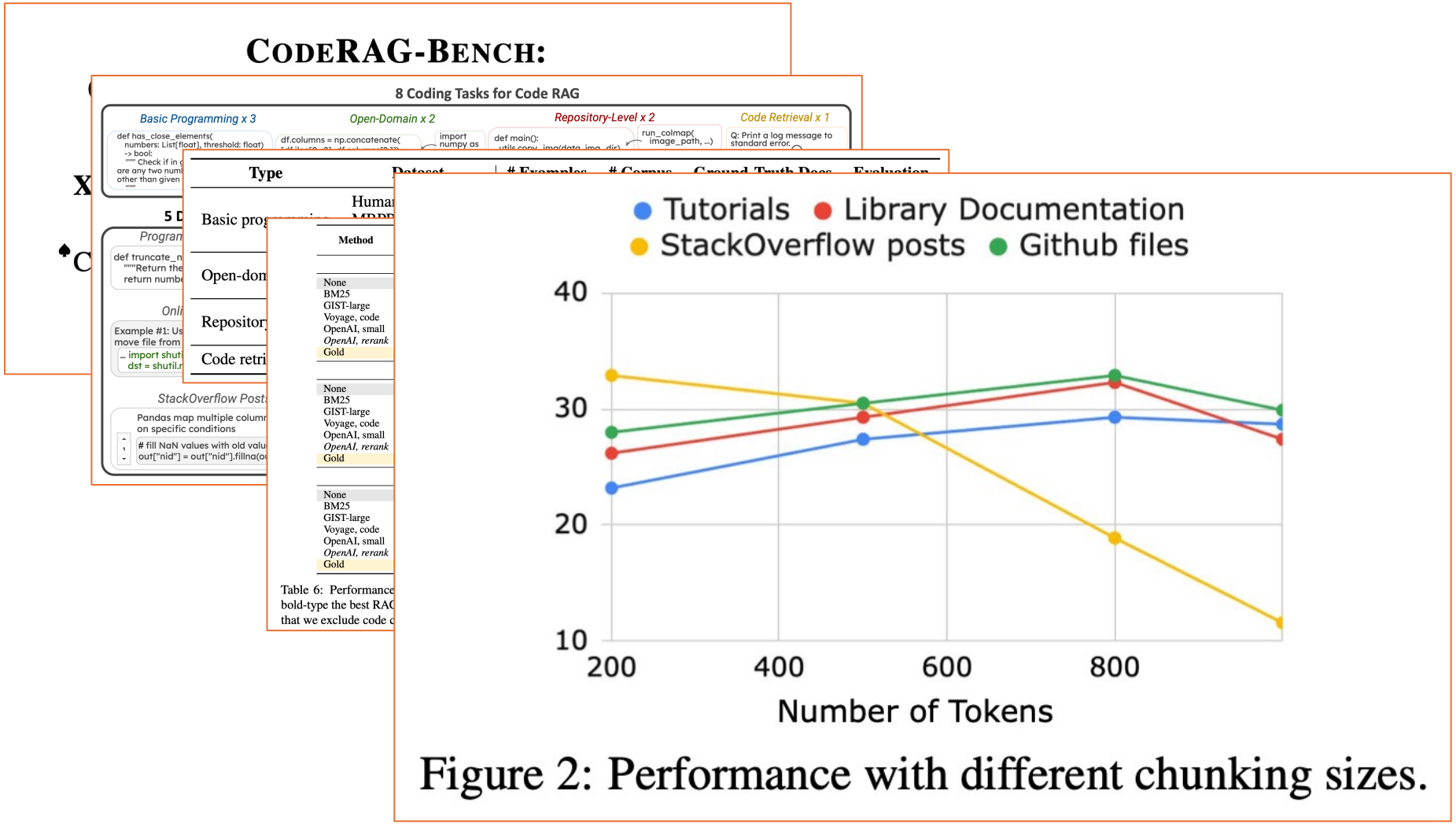
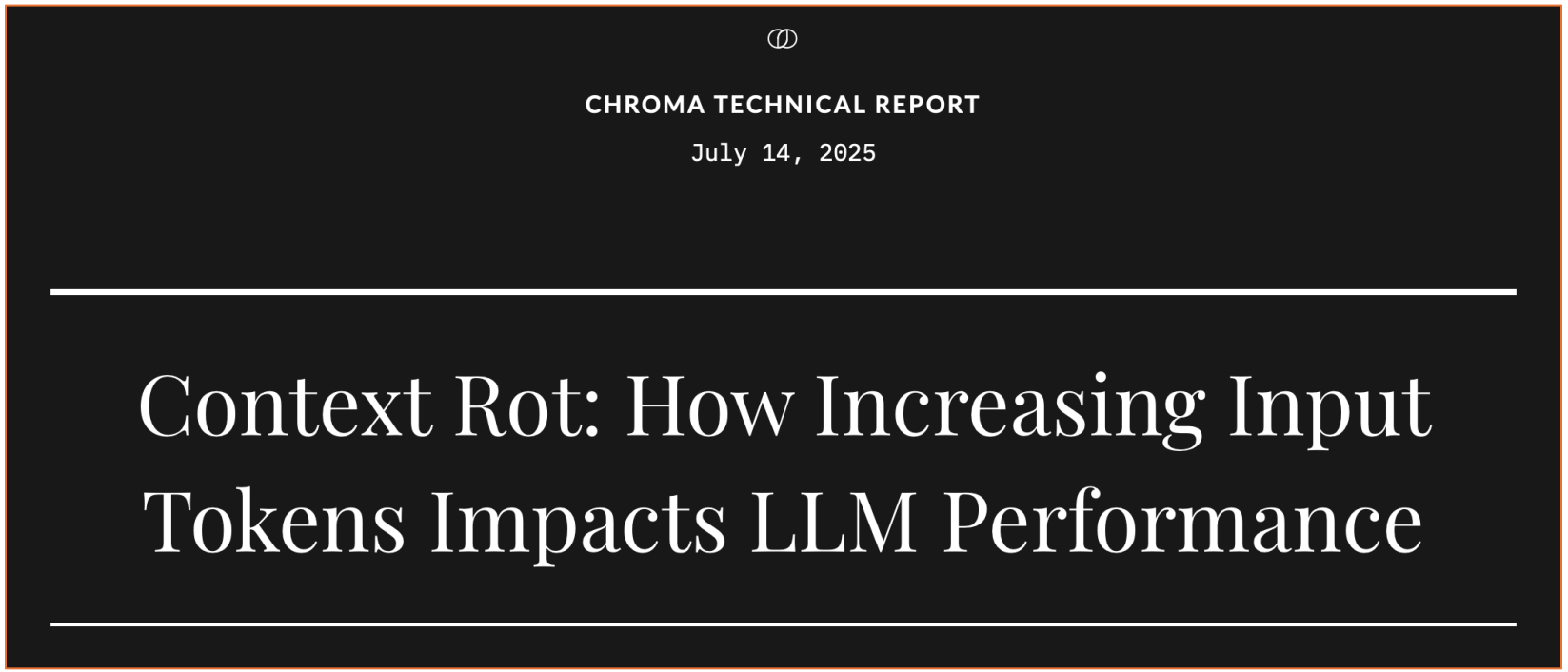


Figure 2: Performance with different chunking sizes.

Note: chunking size is the maximum number of tokens per document; A larger document (e.g., StackOverflow post) will be chunked into multiple smaller documents of chunking size

CodeRAG-Bench: Can Retrieval Augment Code Generation? Wang et. al., NAACL 2025 Findings

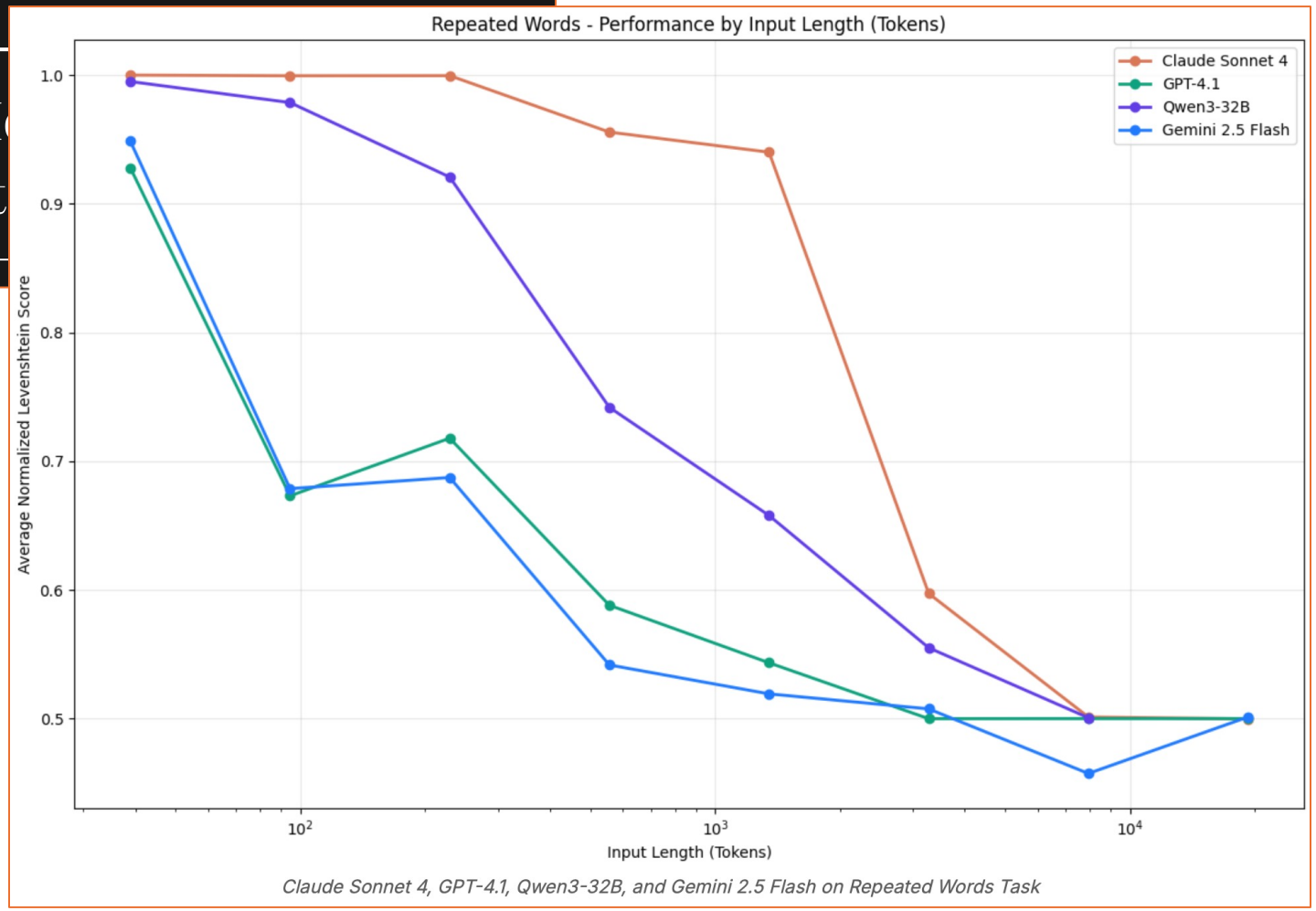
Context Rot


CHROMA TECHNICAL REPORT
July 14, 2025
Context Rot: How Increasing Input Tokens Impacts LLM Performance

CHROMA TECHNICAL REPORT

July 14, 2025

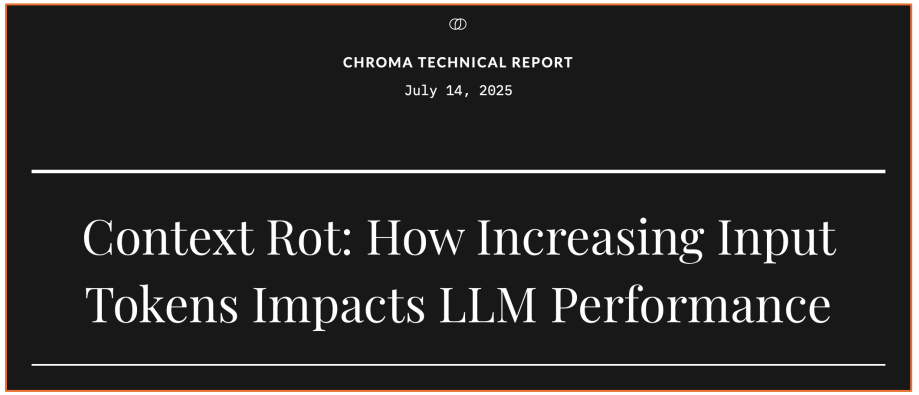
Context Rot: H
Tokens Impact



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

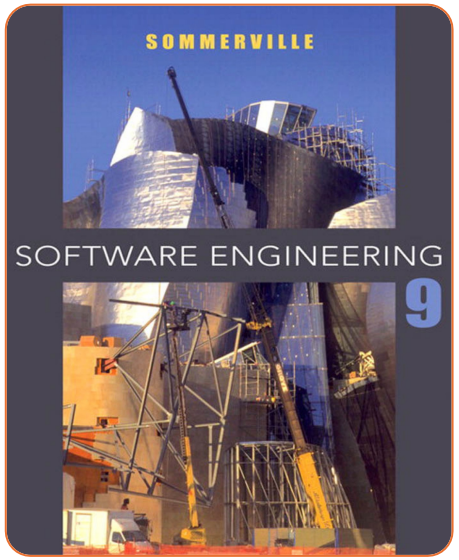
Combating Context Rot with RAG

- … 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…



Software Development Life Cycle (SDLC)

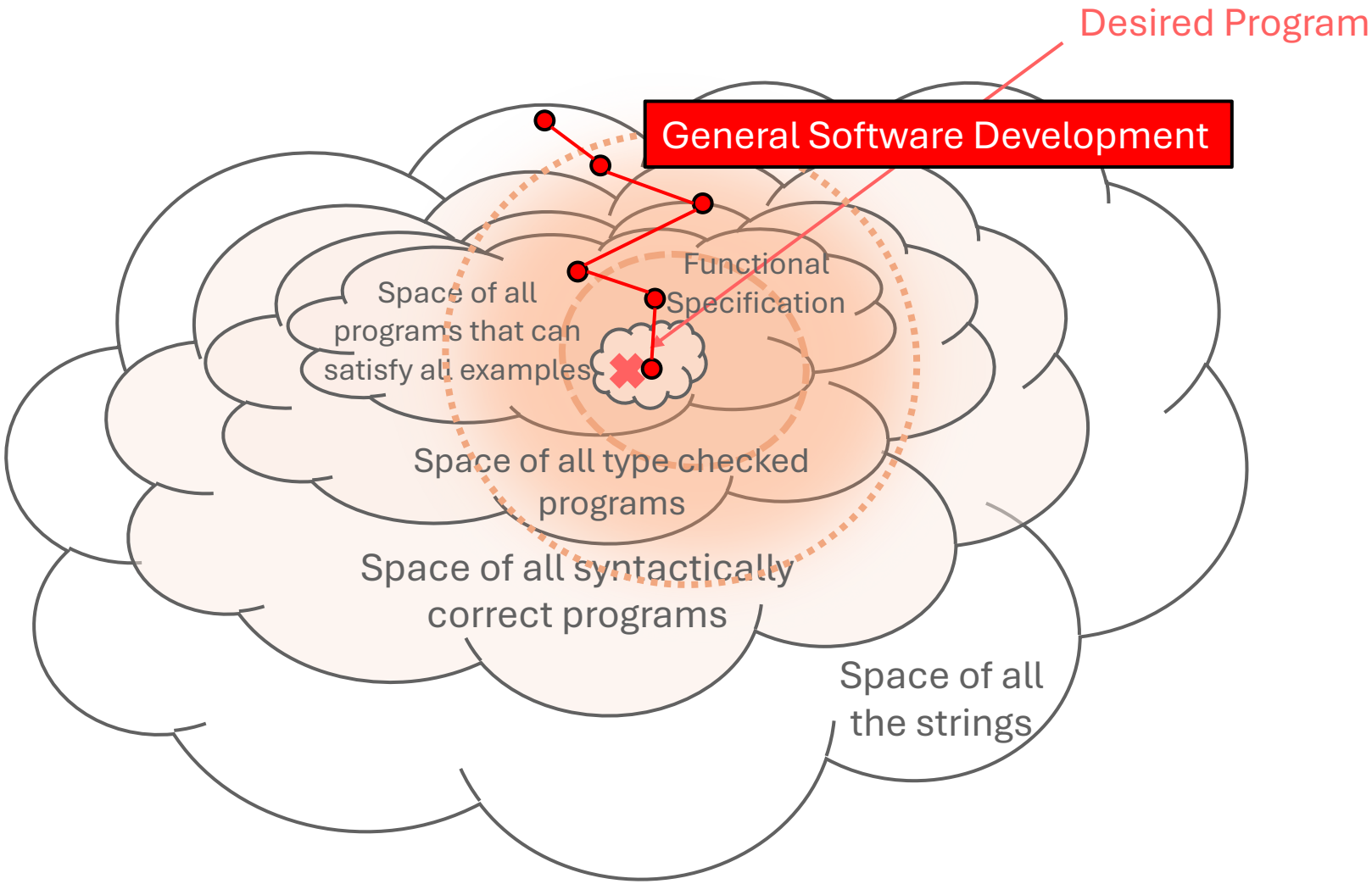
1. Planning → 2. Analysis → 3. Design → 4. Coding → 5. Testing → 6. Deploying



Agentic topics we have not covered (yet!)

- Terminal as tool
- Compiler and interpreter feedback as tool
- Security of agentic frameworks
- Sandboxing

High Level Picture



Logistics – Week 6

- Assignment 2: Evaluating Coding LLMs
  - 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