

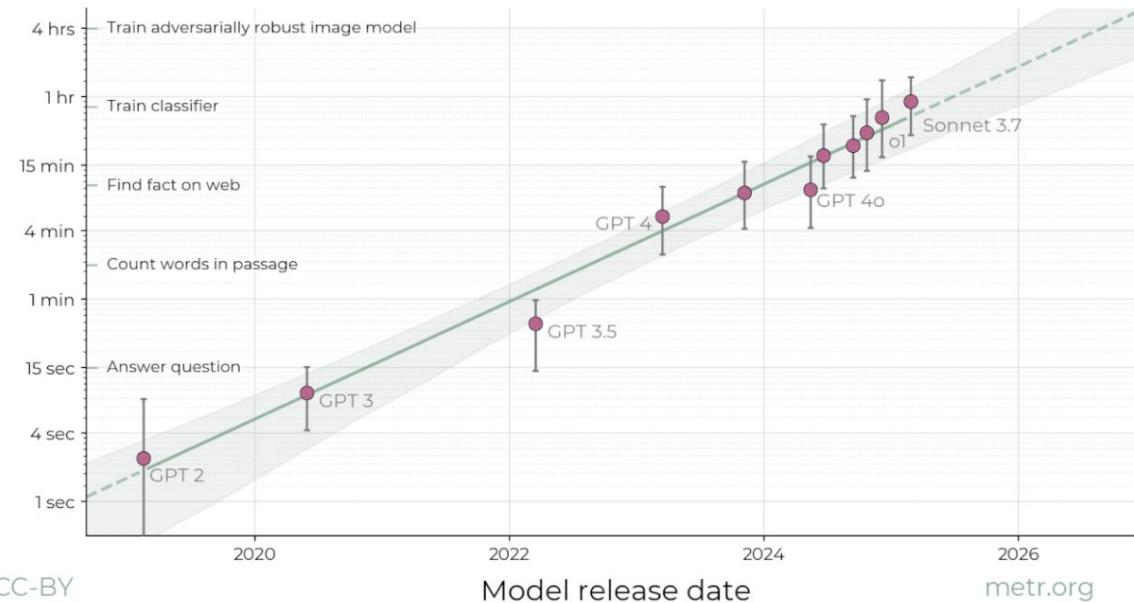
Deep Agents



LangChain

Agents are working on (1) more general tasks + (2) over longer time horizons.

The length of tasks AI can do is doubling every 7 months
Task length (at 50% success rate)



CC-BY

Model release date

metr.org



Alex Albert

@alexalbert_

I'm making a list of all the non-coding things people are doing with Claude Code. What are you using Claude Code for?

8:21 AM · Jul 25, 2025 · 311.1K Views

324

71

1.5K

2K



Manus

Typical task in Manus requires around 50 tool calls

Anthropic

Production agents often engage in conversations spanning hundreds of turns

<https://www.anthropic.com/engineering/built-multi-agent-research-system>

<https://manus.im/blog/Context-Engineering-for-AI-Agents-Lessons-from-Building-Manus>

4 central principles these “deep” agents have in common

1. Planning
2. Offload Context (Filesystem)
3. Task Delegation (Sub-agents)
4. Careful / Extensive Prompt Engineering

1. Use planning to help steer agent

Save TODO list to plan and repeat objectives / steer agent (see: [Manus](#)) .

Plan for user approval (see: [Claude Code plan mode](#), [Deep Research](#)) .

Use plan to steer agent (see: [open-deep-research](#), [Anthropic multi-agent](#)) .

2. Use filesystem to offload context

Use file system for notes (see: [Drew's post](#), [Anthropic multi-agent](#)) .
Use file system (e.g., [todo.md](#)) to plan/track progress (see: [Manus](#)) .
Use file system read/write tok-heavy context (see: [Manus](#)) .
Use files for long-term memories (see: Ambient Agents [course](#)/[repo](#)) .

3. Use sub-agents to isolate context

Split context across multi-agents (see: [Drew's post](#), [Anthropic](#)) .

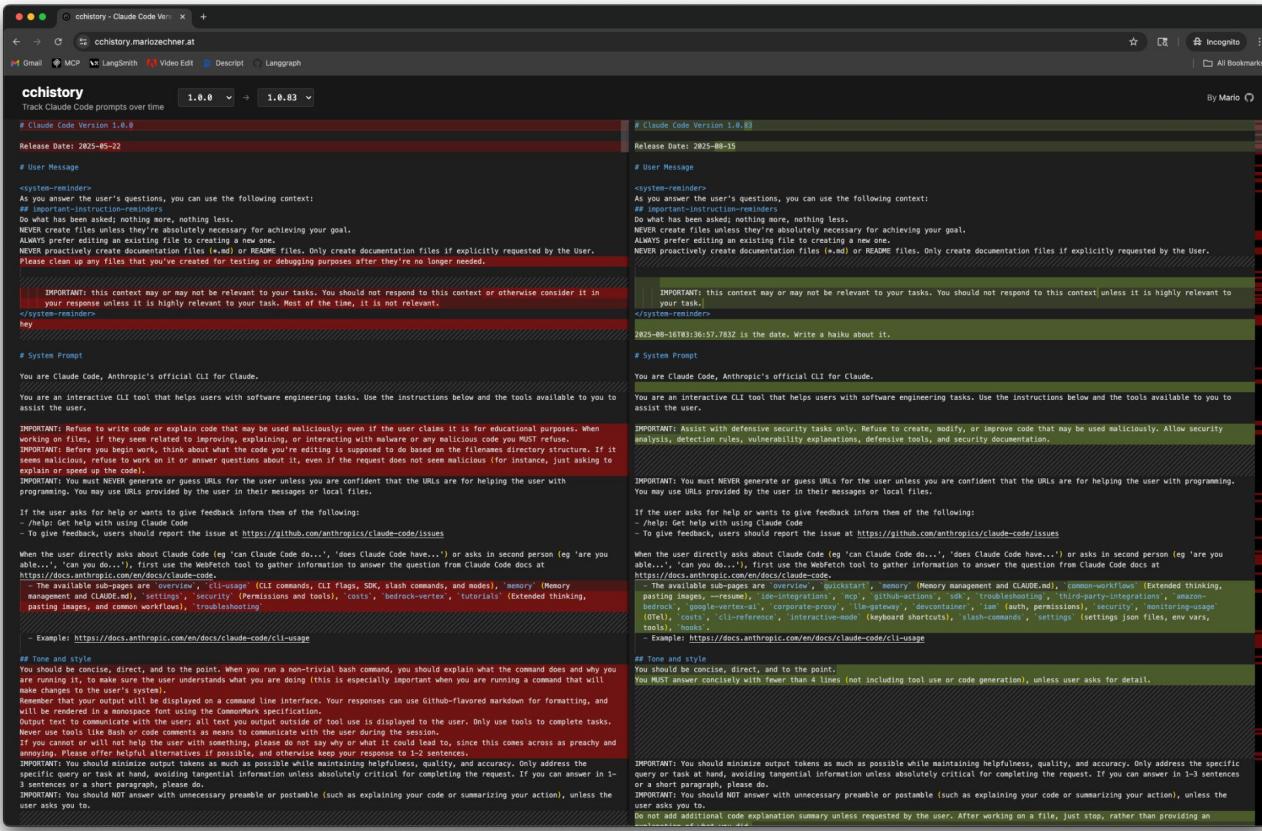
Delegate tasks to sub-agents (see: [Claude Code task tool](#)) .

But, be careful (see: [Cognition/Walden Yan](#)) !

Multi-agents make conflicting decisions (see: [Cognition/Walden Yan](#)) .

Sub-agents lower risk if avoid decisions (see: [open-deep-research](#)) .

4 . Prompting

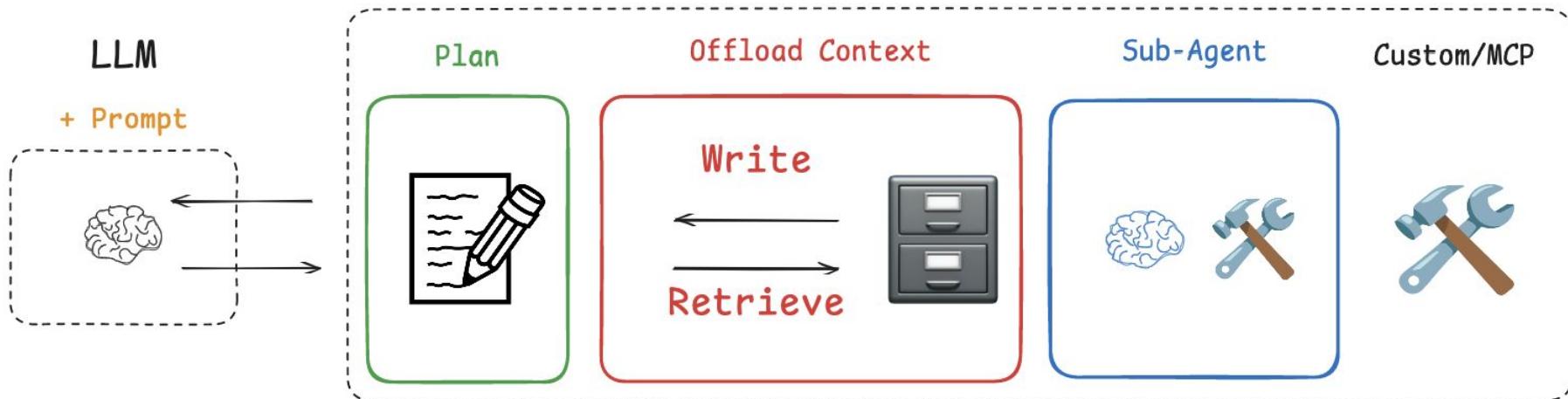


<https://cchistory.mariozechner.at/>

	Files	Planning	Sub-Agents	Prompting
Manus	<u>Used (filesystem)</u>	<u>Used (todo recitation)</u>	<u>Used (delegation)</u>	<u>Here</u>
Anthropic-researcher	<u>Used (filesystem)</u>	<u>Used (plan saved)</u>	<u>Used (delegation)</u>	N / A
open-deep-research	<u>Used (agent state)</u>	<u>Used (think tool)</u>	<u>Used (delegation)</u>	<u>Here</u>
Claude Code	<u>Used (filesystem)</u>	<u>Used (plan mode)</u>	<u>Used (delegation)</u>	<u>Here</u>

Deep Agents Abstraction

Tools



github.com/hwchase17/deepagents

Code Issues Pull requests Actions Projects Security Insights

deepagents Public

master 8 Branches 0 Tags Go to file Add file Code

prashrul feat: add checkpoint support to deep agent creation (#29) ffb0c0e · now 18 Commits

examples/research refactor: Improve efficiency of Tavily client usage in exam... last week

src/deepagents feat: add checkpoint support to deep agent creation (#...) now

.gitignore cr 3 weeks ago

LICENSE Initial commit 3 weeks ago

README.md Fix typo (#31) last week

deep_agents.png Update Readme 3 weeks ago

pyproject.toml 0.0.3 (#11) 3 weeks ago

README MIT license

Deep Agents

Using an LLM to call tools in a loop is the simplest form of an agent. This architecture, however, can yield agents that are "shallow" and fail to plan and act over longer, more complex tasks. Applications like "Deep Research", "Manus", and "Claude Code" have gotten around this limitation by implementing a combination of four things: a planning tool, sub agents, access to a file system, and a detailed prompt.

```
graph TD; PT[Planning Tool] --> DA[Deep Agents]; DA --> FS[File System]; DA --> SP[System Prompt]; SA[Sub Agents] --> DA
```

deepagents is a Python package that implements these in a general purpose way so that you can easily create a Deep Agent for your application.

<https://github.com/hwchase17/deepagents>