13: The open-source ecosystem

The tools under your tools, and the people who maintain them

Why this matters

- You can read the prompts and tool code your agent runs
- Open projects are where harness ideas show up first
- Tools churn; several were archived or shut down in 2026
- AI made pull requests cheap and maintainers' review expensive

Open-source software ≠ open-weight model

Open source publishes the **code**. Open weight publishes the **model**. They're independent.

License	What it allows	Seen on
MIT	Almost anything, with attribution	OpenCode, Kilo Code, OpenHands, T3 Code
Apache-2.0	The same, plus an explicit patent grant	Codex CLI, Gemini CLI, Aider, Cline, goose
FSL-1.1-MIT	Source-available: no competing product; MIT after two years	Crush

A map of the stack

Layer	Examples
Orchestrators and GUIs	Gas Town · Beads · OpenHands Agent Canvas · T3 Code
Harnesses, plus extensions	OpenCode · Aider · Cline · goose · Codex CLI · MCP servers · skills
Frameworks and SDKs	LangGraph · Pydantic AI · DSPy · OpenAI Agents SDK · Claude Agent SDK
Inference runtimes	llama.cpp · Ollama · vLLM · SGLang · MLX
Models	Open-weight or API (modules 03–04)

Evaluation tools (SWE-bench harness, Inspect, promptfoo) measure harnesses and models.

The tools

Harnesses, runtimes, frameworks, and extensions

Open-source coding harnesses (as of November 2026)

Project	Form	Worth knowing
OpenCode	Terminal UI	Works with any model; about 225K GitHub stars
Codex CLI	CLI, IDE extension	OpenAI's open client for its closed models
Aider	CLI	Pioneered the repository map (2023)
Cline · Kilo Code	VS Code extensions and more	Approve each diff; many providers
goose	Desktop app, CLI	General-purpose agent built around MCP
OpenHands · T3 Code	GUIs	Run and supervise several agents

Stars measure attention, not quality.

How to choose

If you want…	Try
One terminal agent that works with any model	OpenCode or Aider
VS Code with approve-each-diff control	Cline or Kilo Code
A general-purpose desktop agent	goose
To run and compare several agents	OpenHands or T3 Code

Before you commit, check the last release date and the open-issue count.

2026 was hard on open-source AI tools

Project	What happened
Roo Code	Shut down May 15, 2026; pivoted to a hosted product
Continue	Archived and read-only after a final 2.0 release
Gemini CLI	Still open, but individual users moved to Antigravity CLI
AutoGen	Archived in favor of Microsoft Agent Framework
Promptfoo	Bought by OpenAI, which says the project stays open
OpenClaw	Creator joined OpenAI; the project moved to a foundation

You can fork an abandoned project. Nobody has to maintain the fork.

Case study: OpenClaw

A personal agent that lives in Signal, Telegram, Discord, and WhatsApp

Date	Name
Nov 24, 2025	Warelay
Dec 3, 2025	CLAWDIS
Jan 2, 2026	Clawdbot
Jan 27, 2026	Moltbot, after trademark complaints from Anthropic
Jan 30, 2026	OpenClaw

It passed 247,000 GitHub stars by early March 2026.

What OpenClaw showed

- Demand is real for agents that live where people already talk
- It's the lethal trifecta in one package: untrusted input, private data, actions
- Defaults are security policy; tens of thousands of instances exposed
- A skill registry is a supply chain; 341 of 2,857 skills flagged malicious
- The fix is sandboxing, least privilege, vetted extensions, confirmations

Inference runtimes: running models yourself

Runtime	Reach for it when
llama.cpp	You want maximum portability, or you're on a laptop
Ollama	You want the easiest local setup: `ollama run <model>`
vLLM	You're serving a model to a team or a product
SGLang	You serve agent workloads with lots of shared prompt prefixes
MLX	You're on a Mac and want speed or local fine-tuning

Most harnesses can use any of these through an OpenAI-compatible endpoint.

You may not need a framework

The heart of the module's 30-line agent loop (abridged):

```
for turn in range(30):                          # a turn budget
    resp = client.messages.create(model=MODEL, max_tokens=4096,
                                  tools=TOOLS, messages=messages)
    messages.append({"role": "assistant", "content": resp.content})
    if resp.stop_reason != "tool_use":
        break                                   # the model is done
    messages.append(tool_results(resp)) # run each tool, ask y/N first
```

A loop, a framework, or a harness?

Use	When you need
A plain loop	One agent, a handful of tools, short sessions
A framework or SDK	State across restarts, tracing, many tools or agents, approval steps
An existing harness	"A coding agent, but scripted"

Frameworks: LangGraph · LlamaIndex · DSPy · Pydantic AI · CrewAI · Mastra · smolagents

Lab SDKs: OpenAI Agents SDK · Claude Agent SDK · Google ADK

Extensions: MCP servers and skills

- Reference MCP servers are educational, "not production-ready solutions"
- Official servers: GitHub, Playwright (Microsoft), Chrome DevTools (Google)
- Find published servers in the official MCP Registry
- Skills: anthropics/skills, obra/superpowers, the Agent Skills spec
- Extensions run with your agent's permissions; they're dependencies
- Read before installing, pin versions, grant the narrowest credentials

Orchestrators: Gas Town and Beads

Gas Town (Steve Yegge, January 2026) runs 20–30 coding agents at once

Role	Job
Mayor	Takes your request and assigns work
Polecats	Worker agents
Rigs	Wrap repositories
Witness	Watches for stuck agents
Refinery	Batches and verifies finished work before merging

Beads is a graph issue tracker, backed by Dolt, that agents share as memory

OpenClaw and Gas Town

> **Bryan's take:** OpenClaw had a couple of issues, but it was Gas Town that really did not work for me.

The maintainers' side

AI made contributions cheap and review expensive

curl ended its bug bounty

Measure	Value
Confirmed vulnerabilities, all time	87
Paid out	More than $100,000
Share of reports that were real, earlier years	Above 15%
Share of reports that were real, 2025 on	Below 5%
Bounty ended	January 31, 2026

"Not even one in twenty was real." (Daniel Stenberg)

Project AI policies range widely

Project	Policy	In short
Gentoo	Ban (April 2024)	No AI-assisted content: copyright, quality, ethics
QEMU	Decline	Rejects contributions believed to derive from AI output
LLVM	Human in the loop	Review it all, label it, write your own PR text
Linux kernel	Human accountability	"AI agents MUST NOT add Signed-off-by tags"
Ghostty	Disclose and understand	Disclose all AI use; newcomers need a vouch

The 2026 trend: away from bans, toward disclosure and accountability.

Contributing with AI: etiquette

- Read `CONTRIBUTING.md` and the AI policy, then follow them
- Understand every line; explain it without the AI open
- Reproduce bugs yourself, on a real build
- Keep PRs small; run the project's tests and linters first
- Write the description and replies yourself; disclose AI use
- Never let an agent post for you; accept a no

For students: contribute to learn

- Pick a project you use; read its rules first
- Use AI to understand the code, not to submit it
- Claim an issue; some "good first issues" are AI-free by design
- Reproduce the bug by hand; write a failing test first
- Rework any agent draft until you could have written it
- Keep a log of what you learned; that's your evidence

Review before the maintainer does

> **Bryan's take:** Having several agents or models review each other's work is valuable. That applies before you open a pull request on someone else's project too. Run a second reviewer over your diff before a maintainer has to.

Try it

On your own (2–3 hours), read a harness, then write one:

1. Clone OpenCode, Aider, Cline, or Codex CLI; find its system prompt and tools
2. Run the 30-line loop against a toy repo with a failing test
3. Add one idea from the real harness; measure it on three tasks

As a class, take the maintainer's chair: review ten mixed PRs, time each, and spot the unreviewed AI output.

Discussion

1. Is an open harness that only works well with one closed model really "open"?
2. How should a team decide whether to depend on an open-source AI tool?
3. Gentoo bans AI contributions; LLVM allows them with review. Which would you pick?
4. Who should bear the cost of filtering AI-generated reports?

Read more

- Module 13: The open-source ecosystem
- Building effective agents (Anthropic, December 2024)
- The end of the curl bug bounty (Daniel Stenberg, January 2026)
- LLVM AI tool use policy