MULTIAGENT

A weekend experiment with autonomous engineering agents.

Gianluca Guida, 4 June 2026

Who am I?

Long story short.

- Gianluca
- Founding Engineer @ Ainekko (nekko.ai)
- I currently live in two cities: Cambridge and London, England
- My office is technically on other side of the hemisphere: San Francisco (CA)
- Lived in: Bari, Palo Alto, Milan, Amsterdam, Cambridge and London.
- Ex: Apple, HP, Rivos, XenSource + other startups. Intern @ VMware (2006).

Why am I here?

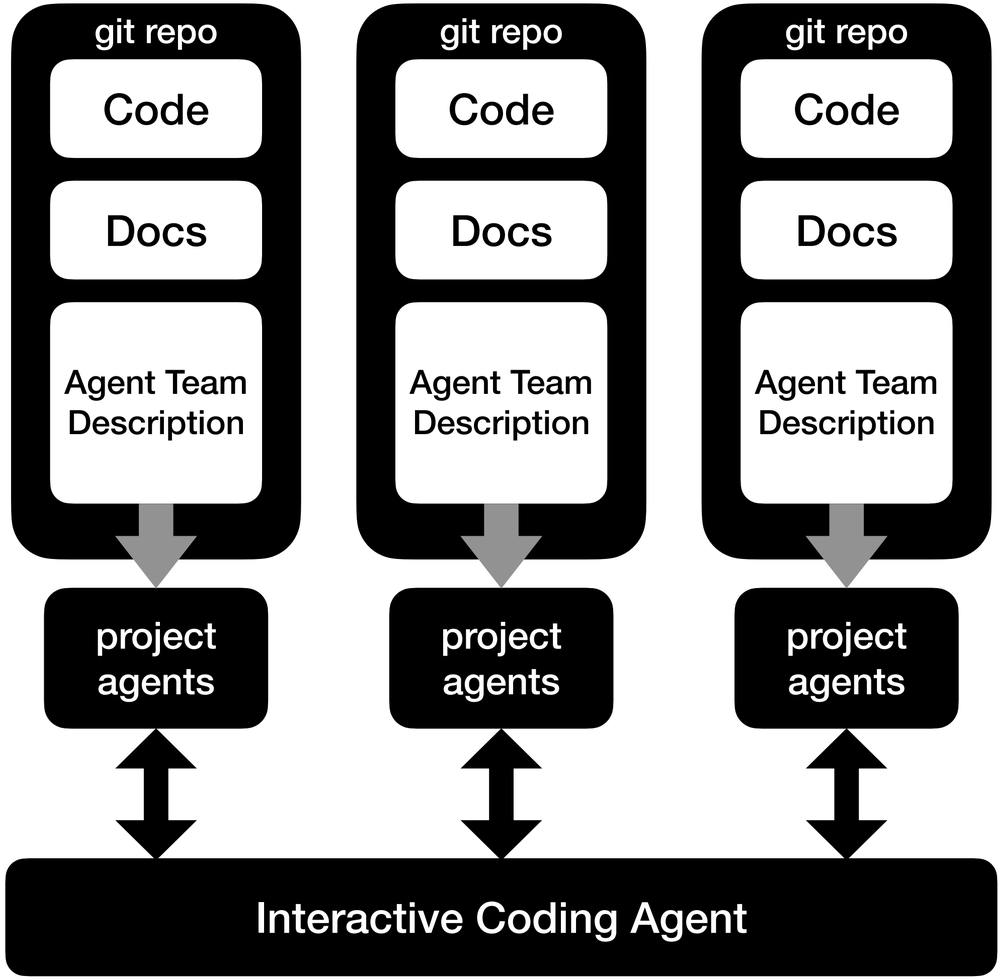
There’s this AI thing going on.

- Responsible for software in a Open-Source AI HW startup during the AI revolution. Some of the questions I have to answer:
  - How to structure **work** in a world where **coding agents are a personal tool** of every software engineer?
  - At what level can **engineering** be **automated**?
- Ainekko has need to open source pre-existing HW and SW IP
  - HW IP (RTL) is tricky, requires modernisation of codebase.
  - Decided to experiment using **Agentic Workflows** for the task.
  - This later became OpenHW’s CORE-ET: https://github.com/openhwgroup/core-et
  - **multiagent** is the result of the need to automate the effort.

multiagent: an overview

A multi-project system for heterogeneous agents

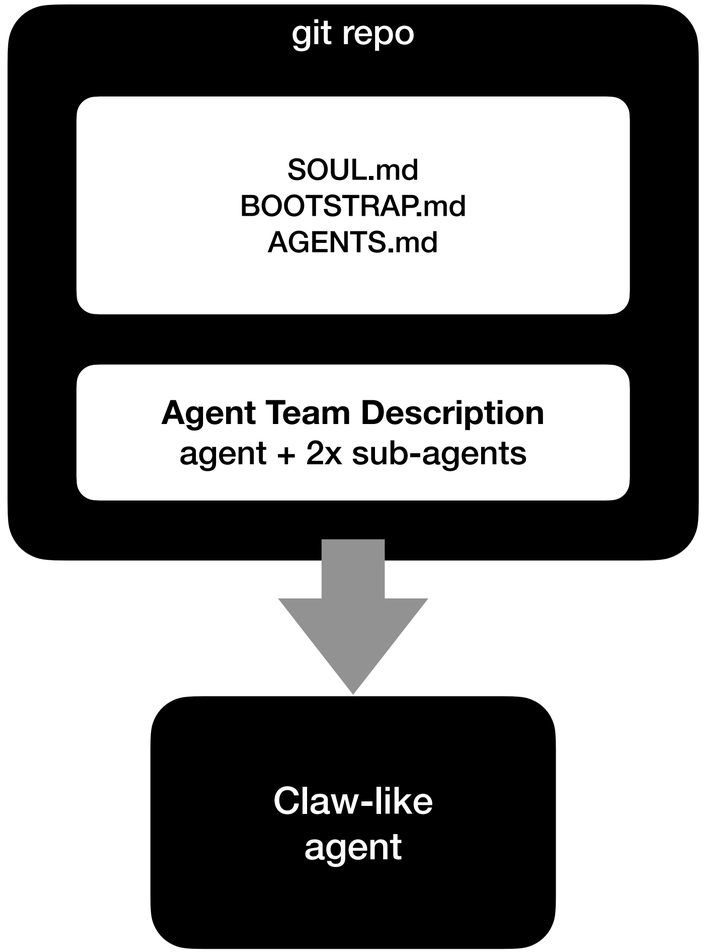
- **multiagent** is a personal project born out of need, not design.
- Iterative development and poorly documented.
- Different projects require specialised agents.
  - Agent workflow and definitions (`<role>.md`) *lives* in the git repository and *evolves* with the project.
- **multiagent** runs project agents in a project specific *container*.
- The interactive coding agent creates **tasks**, the agent team works on the task until completion.



Repoclaw: a multiagent claw-like agent

Clone repo ↦ claw agent.

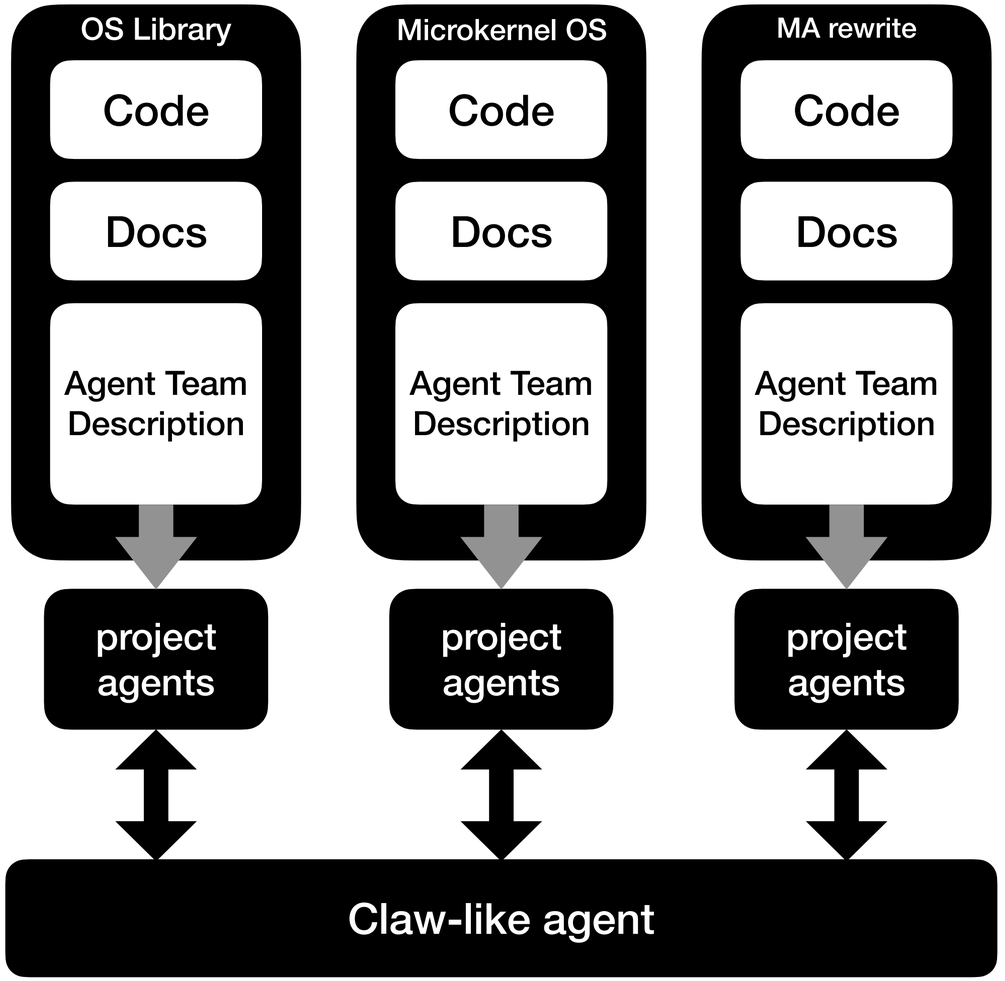
- To explore with autonomous task handling, openclaw seemed the right choice.
  - Simple, efficient, *proactive*.
- **repoclaw:** a multiagent agent team implementing a claw-like system.
  - Testing interactive agents support.
- Similar to Git Agent Protocol, but based on multiagent:
  - Each git local clone is a different agent.



multiagent claw experiment: setup

Handling pre-AI era projects and new projects autonomously

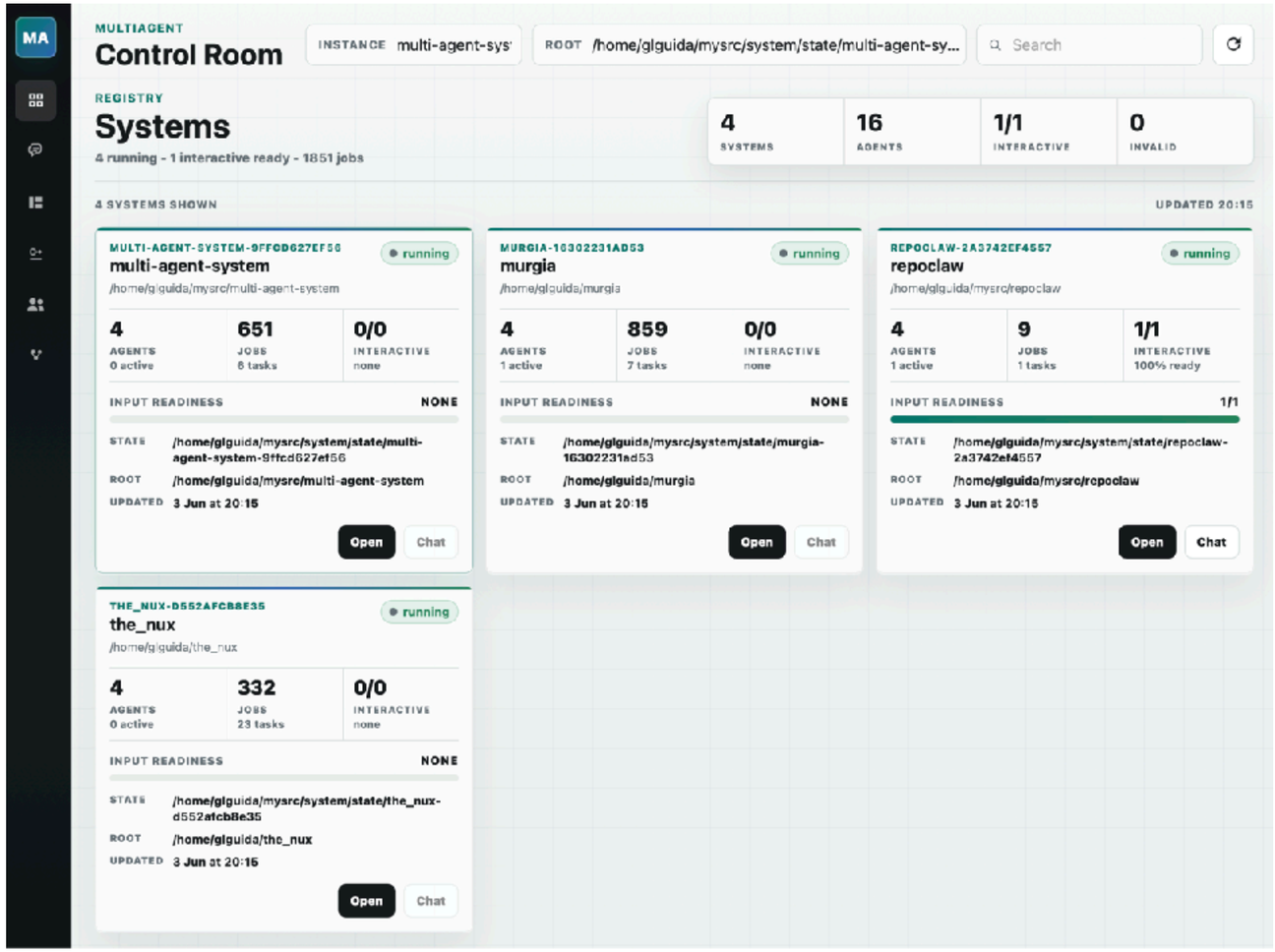
- Past weekend hack.
- A system with three projects:
  - My OS kernel library.
  - My microkernel OS based on the library.
    - Request: continue work and improve code quality, respecting architecture.
  - A rewrite of multiagent from scratch.
    - Request: “reimplement, cleanup and expand multiagent”.
- Default multiagent team: **planner ↦ implementer ↦ reviewer ↦ committer**
- Model: **gpt-5.5 xhigh**



multiagent claw experiment: results 1/3

That’s a proper weekend hack!

- Left alone after setting it up, checked after arrival in Venice.
- Hit weekly limit (codex pro max) after three days.
- 38 tasks planned and completed
- split into 1842 jobs
- 100+ commits, with documentation, test and reviews.



multiagent claw experiment: results 2/3

claw as an OS developer: 👍

- Cleaned up build system
- Implemented IOMMU support
- Added tests (and fuzzing) to the kernel interface!
- Started implementing AHCI (disk) driver before running out of tokens.
- Test-driven approach created working code, well tested.
- Correctly handled dependencies of tasks between two projects (library and microkernel) and respected existing architecture.

multiagent claw experiment: results 3/3

claw as a free-style implementer from scratch: 👎

- **Good**:
  - Cleaned up what needed to be cleaned up
  - Added impressive features: federation (via HTTP+MAC and ssh) to connect system across network securely.
  - Everything was extremely well documented.
- **Bad:**
  - Extreme test-driven methodology drifted the project in a meta project about “proofs” and “evidence” of working functionality.
  - At End of Tokens, the project was mostly a mean to run complex tests than to actually run a multiagent system. The actual goal of having a system running become secondary to having a testing framework.

multiagent claw experiment: lesson learned

Calm down, claw!

- Until it was left alone after initial setup, the system behaved properly, except for drifting due to lack of specification.
- My claw-like “personality” became a bit too obsessed and overreacting
  - Extremely *proactive*
  - Felt like an *anxious person high on caffeine*
- It managed properly the dependencies, understood the context (can read the code to understand, but dispatched tasks to a specialised team).
- Poorly documented system created some confusion: claw-like agent just did not understand that each project ran in its own container, and kept panicking about not being able to see the processes. This is a multiagent limitation, should be fixed with proper skills/documentation.

Conclusion

This was fun and surprising.

- The autonomous behaviour suprised me.
- Still imperfect, will require more iterations.
- Having the ability to define per-project agents allows me to think about system interaction rather than relying on a complex single coding agent.
- The personality of the interactive agent will require major work.

Thank you!

Questions?

About me: https://tlbflush.org

My github: https://github.com/glguida

- You will find multiagent, repoclaw and various other experiments there.