What Open Source Was Supposed to Be

February 8, 2026 · by Mark Bartlett

Download this post as PDF

In 1983, Richard Stallman started the GNU Project with a radical idea: software should be free. Not free as in beer—free as in freedom. The freedom to run it, study it, modify it, share it. Software as a commons, not a product.

For a while, it worked. Linux ran on everything from supercomputers to routers. Apache powered most of the web. MySQL stored the world's data. Regular people could download the same tools that Fortune 500 companies used. The playing field wasn't level, but at least everyone was on it.

That's not what open source means anymore.

What It Became

Somewhere along the way, "open source" became a business strategy instead of a philosophy.

The playbook is familiar now: release a project under a permissive license, build a community, accept pull requests from volunteers, then wrap the good parts in a paid cloud service. Your contributions become their product. The code is technically open. The value extraction is closed.

MongoDB did it. Elastic did it. Redis did it. HashiCorp did it. The pattern is so common it has a name: open core. The "core" is open. Everything useful costs money.

Even when the code stays genuinely open, the infrastructure doesn't. You can read every line of Kubernetes. Good luck running it without a cloud provider. You can fork TensorFlow. You'll still need Google's TPUs to train anything serious.

Open source won. And somehow, we lost.

The AI Version Is Worse

OpenAI is "open" in name only—they dropped the pretense years ago. But even the genuinely open AI projects have a problem.

Meta releases Llama. The weights are free. The license is permissive. You can download the whole thing right now. Victory for open source, right?

Except Llama 70B needs 40GB of VRAM to run at Q4 quantization. That's dual RTX 3090s, minimum. The 405B model needs a datacenter. You can read the weights, but you can't use them without spending thousands on hardware or renting GPU time from the same cloud providers that open source was supposed to free us from.

The gap between "released weights" and "actually runnable by normal people" isn't a technical detail. It's a moat dressed as a gift.

Mistral, Qwen, DeepSeek—same story. The models get bigger. The hardware requirements grow. The weights are "open" but the ability to run them concentrates in fewer and fewer hands.

Meanwhile, the real action happens behind API walls. GPT-4's weights will never be released. Claude's won't either. The most capable AI systems are black boxes you rent by the token. Your prompts, your data, your use cases—all flowing through servers you don't control, under terms of service that can change overnight.

This is what open source looks like in the AI era: you can read the research papers but not replicate them. You can download the weights but not run them. You can build on the APIs until the pricing changes or the model gets deprecated or your use case gets flagged by a content policy written for someone else.

The Real Test

Here's a simple filter for whether something is genuinely open:

Can a student in Lagos run it? Not "could they theoretically if they had different hardware"—can they actually run it on what they have?

Can a maker in Berkeley build on it without asking permission? Not "is the license permissive"—can they actually use it in ways the original creators didn't anticipate?

Can it survive if one company disappears? Not "is the code archived somewhere"—would the project keep working if the primary maintainer got acqui-hired or pivoted to enterprise?

Most "open" AI projects fail at least one of these tests. Many fail all three.

The four freedoms Stallman defined—run, study, modify, share—assumed you could run the software. That was the easy part in 1983. A C compiler didn't need a datacenter. The hard part was getting access to the source.

Now the source is everywhere. The ability to run it is the scarce resource.

What Reclaiming Looks Like

The hardware already exists. There are millions of "obsolete" GPUs in drawers and closets—RTX 3060s from 2021, GTX 1080 Tis from mining rigs, M1 MacBooks that still work fine. A 3060 with 12GB of VRAM can run a 14B parameter model. A ThinkCentre you can buy for $100 on eBay can serve embeddings or run a small model on CPU.

The problem isn't hardware. The problem is coordination.

A single RTX 3060 can't run a 70B model. But five of them together have 60GB of VRAM—more than an RTX 4090. The question is whether they can work together effectively.

This is the thesis: **coordinated cheap nodes can beat single expensive ones.** Not in raw throughput—a 3090 will always be faster than three 3060s for a model that fits. But for models that don't fit, for workloads that can be parallelized, for resilience against single points of failure—the swarm has advantages the monolith doesn't.

The network is the computer. But your network. Your computers. Hardware you own, running software you control, storing data that never leaves your premises.

No API keys to revoke. No rate limits. No terms of service that change at 2am. No "we're pivoting to enterprise" emails. No vendor lock-in because there's no vendor.

This isn't nostalgia for a simpler time. It's recognition that the original open source vision was right, and the current implementation has drifted. The answer isn't to reject AI or retreat to older technology. It's to apply the original principles to the new context.

---

mycoSwarm

This is why I'm building mycoSwarm.

Not as a product—there's no company behind it, no roadmap to monetization, no enterprise tier waiting in the wings. As a proof of concept that the original vision still works.

The name comes from mycelium—the underground fungal networks that connect forests. Mycelium doesn't have a center. It doesn't force its way through soil; it finds the path of least resistance. Individual nodes are simple. The intelligence emerges from connection.

The idea is simple: any device with a CPU or GPU can join the swarm. A 3090 workstation becomes an "executive" node for heavy inference. A ThinkCentre becomes a "worker" for embeddings and document processing. A Raspberry Pi becomes a coordinator. They find each other automatically on the local network, advertise their capabilities, and route tasks to whoever can handle them best.

Two old laptops can do what neither could alone. A gaming PC and a mini PC can collaborate on workloads that would overwhelm either one. The barrier to entry is "hardware you already own."

It's early. The discovery layer works—nodes find each other in under a second with zero configuration. The capability detection works—each node knows what it can run. The API layer works—nodes can query each other's status and submit tasks. The actual distributed inference is next.

Maybe it works. Maybe it doesn't scale. Maybe the coordination overhead eats the theoretical gains. That's what the project is for: to find out.

But the question it's asking is worth asking. Can we build AI infrastructure that actually embodies open source values? Not "open" as a marketing term. Open as in freedom.

Can we reclaim what open source was supposed to be?

📚 **More on this topic:** Best Local Models for OpenClaw · Budget Local AI PC · Multi-GPU Local AI

mycoSwarm: GitHub
