Two machines.
Three tiers.

The system behind Tuesday's race-day fuel chart.

Frontier reasoning where it matters. Local extraction where it doesn't. Filesystem handoff between them. 11 weeks of training data, three race-day artefacts, one stack.

swipe →

The constraint

LOCAL ONLY

Raw inputs

sweat test (614 mg/L)
PF&H protocol
training run logs
Andy's coaching notes
GI history
race profile

Private. Personal. Not going through a third-party API.

FRONTIER ONLY

Synthesis

reconcile worker outputs
build decision trees
cross-reference domains
spot contradictions
verify against constraints
produce final artefacts

Wide-scope reasoning. Where frontier models earn their place.

The split is not arbitrary. Each side has a job the other can't do well.

02 / THE PRINCIPLE

Frontier where reasoning matters.

Local where token volume matters.

Extraction is high-volume, narrow-scope work. Reading 40+ sources, pulling structured JSON out of each. Local models do this fine with tight prompts.

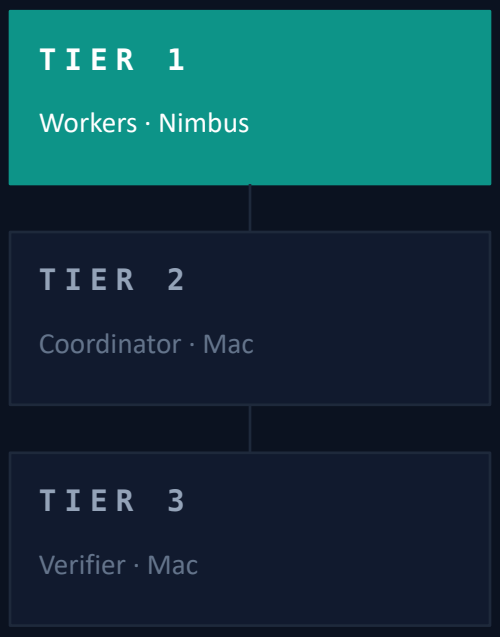
Synthesis is low-volume, wide-scope work. Reconciling worker outputs, spotting contradictions, building decision trees. This is where frontier reasoning earns its place.

The cost split is dramatic. Tokens at Tier 1 are effectively free — local hardware, marginal cost ~$0. Tokens at Tier 2 and 3 are budgeted and used deliberately.

THE RULE OF THUMB

If the task is read 50 things and output 50 structured records, run it locally. If the task is read 50 records and decide what they mean together, run it on a frontier model.

The Nimbus

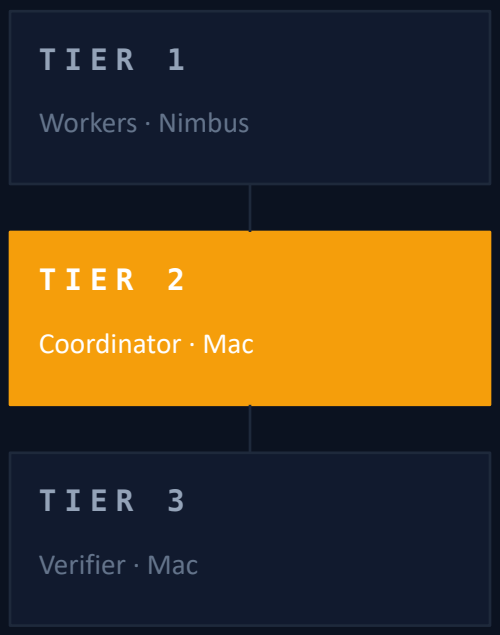


machine	Nimbus PC
gpu	RTX 5060 Ti · 16 GB VRAM
runtime	Ollama
model	qwen2.5-coder:14b
quant	Q4_K_M (~9 GB)
context	8192 tokens
parallel	3 workers concurrent

Each worker gets a narrow job: extract a single domain. Carb protocol. Sodium protocol. Failure modes. Hydration. Each writes structured JSON to /extracted/. The 14B at Q4_K_M is small enough to run three workers in parallel on 16 GB — and good enough that the structured outputs are clean.

TIER 2 / COORDINATOR

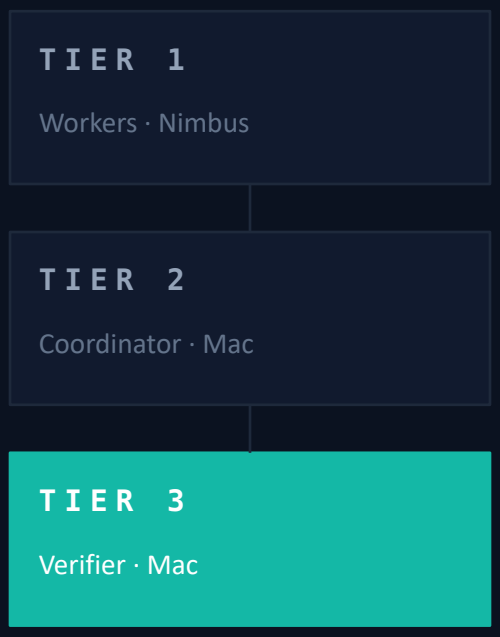
The Mac Mini



machine	M4 Mac Mini · 16 GB
runtime	Claude Code
model	Claude Sonnet 4.6
inputs	/extracted/*.json
outputs	/synthesised/*.md
job	synthesis · cross-reference · decisions

Reads the JSON the Nimbus produced. Reconciles inconsistencies between workers. Builds the decision trees. Cross-references against the other domains — hydration affects fueling, fueling affects mindset, mindset affects pacing at km 80.

Fresh session.
Cold context.



machine	M4 Mac Mini · 16 GB
runtime	Claude Code · new session
model	Sonnet 4.6 · escalates to Opus 4.x
inputs	/extracted/ + /synthesised/
outputs	/artefacts/*.md
job	verify · catch what coordinator missed

The verifier is a fresh Claude Code session. No memory of the coordinator's reasoning. It reads both the workers' raw outputs and the coordinator's synthesis, and produces the final artefacts. Cold context is the point — the verifier catches what the coordinator missed because it's not committed to the coordinator's frame.

Filesystem, not API.

`/extracted/*.json` → `/synthesised/*.md` → `/artefacts/*.md`

Machines talk over SSH. Workers write JSON. rsync pulls files to the Mac. Coordinator and verifier work on local files. Final artefacts land in /artefacts/.

No hidden state. Every step is inspectable. Every failure is reproducible. The system is debuggable the way a Makefile is debuggable.

An API integration would have been faster to write. The filesystem version is faster to trust.

— and faster to debug when it gets things wrong. Which it does.

Extraction is harder than it looks.

v0.1 · FIRST PASS

240 g/hr

8 gels every hour
for 16 hours

3,840 g total

not survivable

v2.3 · CONSTRAINT LAYER

70–90 g/hr

2–2.5 gels per hour
± terrain · ± temperature

~1,250 g total

proven in 3 ultras

The model knew the literature. It cited the studies — elite cycling, short duration, lab conditions. It did not know my gut.

Fix: a constraint layer. My actual race logs added to the prompt. Hard cap on intake at what I have actually digested in real ultras.

The model knew the field. It didn't know the body.

Three tiers.
Three trust profiles.

WORKERS

Cheap, fast, narrow. Verify their structured output programmatically — schema, ranges, sanity. Trust nothing they say in prose.

COORDINATOR

Synthesises. Make every intermediate file readable. Show your work. The point isn't autonomy — it's auditability.

VERIFIER

Different session, different role, fresh eyes. Cold context is the actual feature. The verifier exists because the coordinator can talk itself into things.

The architecture isn't about the models. It's about where you spend trust.

Where are you drawing your tier boundaries?