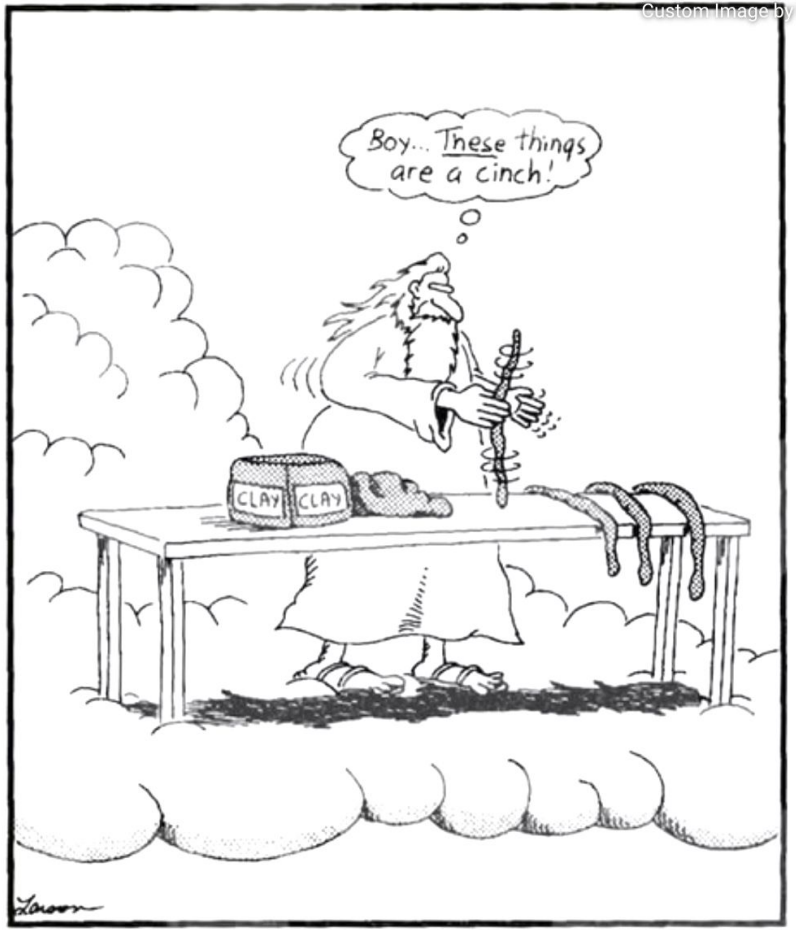
Learnings from building a Strategy-first Agentic Network

TechDivision

BE STRONG. BE REAL. BE DIGITAL.

Warning: Claude Code might cause a God Complex



God makes the snake

The Problem

What happens when you build too many agents (or too few)?

- Agent 1 does X. Agent 2 does the same. Nobody knows why.
- No memory. No coordination. No context.

You've reinvented fire...

...and everything is burning.

The Solution

Relate everything to your Strategy!

A central coordination layer that all agents align to

- The strategy core knows everything about your company and where you want to go.

  It is home to an orchestrator that chooses which sub-agent to use for the task at hand.

- It relays information between agents, agents never talk to each other directly

- It controls input and output and has access to memory

Technical Part: Scaffolding

Don’t start by building features, build structures first!

Step 1: Sessions
- Every agent has a session, no global state
- `session_id` = agent's identity

Step 2: Agent Registry
- Where does the system know which agents exist?
- Registry as Single Source of Truth

Step 3: Agent Persistence
- What survives a restart?
- Redis: SHORT_TERM (24h) / WORKING (7d) / LONG_TERM (permanent)

Step 4: Message Bus
- Agents don't talk directly to each other
- Priority routing: 1–10, urgent skips queue

= 847 messages/week, 12ms latency

Strategy: What really matters

What are the basic facts and the plans for the future.

Give the System as much real-time and verified context as possible.

- get data from your ERP, CRM, BI-Tools, your domain software
- Feed it quality, verified content, not just everything on drive Z, hoping for the best
- Create Agents for the main tasks: Research, Evaluation, Scenarios, Condensation

Pro-Tipp: get a higher evaluation agent in the game with your personal strategy-gods:

- Mine are on the “Strategy Council”: 4 widely published (!) management perspectives:
- Peter Drucker (MBO), Fredmund Malik (Systemic), Stafford Beer (VSM), Jim Collins (Flywheel)
- Recommendations always pass through this filter
- Result: strategically coherent outputs, not just data dumps

Humans understand Humans

At least better than Bots!

Why?

- **Names convey methodology and expectation**
- **Team intuitively understands what an agent does**
- **AI gets a personality, not just a function**
- **Agent inherits authority from the name**

My Examples:

- **Strategy Council** → Drucker, Malik, Beer, Collins
- **Code Inspector** → "Doubleknuth it!"* (Donald Knuth, The Art of Computer Programming)
- **Chris Walker (B2B Marketing shark)** → after provocative influencer

...When an agent has a name, people treat it differently

Performance & Costs

Not everything needs an Einstein!

Speed matters: A slow system means you and the machine sit around waiting - a lot!

Assign specific modells to the agents: we will spend less than 100€ on the agent networks operation per year, instead of close to 3.000€ (estimates) because we use Opus only for the council, the rest runs on Sonnet and even Haiku, depending on the agent's task

Avoid instead of fixing: Add a github account with a CI Pipeline to test each change, document everything, have the system create software tests for every function you add. Every problem you don't build into your system thats cheaper to avoid than to fix

Outlook

Whats next for our strategy network?

- **A Webinterface/GUI!**
- **Less Agents!***
- **Distributed working mode!**
- **Interchangeable company contexts!**

***Or more?**

Takeaways & QR Code

Do try this at home!

- **Start with sessions & registry — not with 26 agents**
- **Strategy Core first, then build agents around it**
- **Name central agents after people → bias and methodology immediately clear**
- **Hybrid model: Sonnet for 95% of the work**

And stay out of trouble..

…by not get carried away by Claude giving you superhuman vibes

Claude Code tries to do what it can to max out your Usage and make you buy more credits

- Always ask if there is a standard library or Open Source solution you can use
- Always go for the lesser version of a big feature, you will have less hassle debugging it
- Grant all the rights to Claude Code it needs to perform the task at hand beforehand
- Work as if tokens would be 10x the price tomorrow
- Invest in your work environment (Claude.md, Credentials, memory.md etc)

NGO much?

Pro Bono/Open Source Bonus Track!

Think outside of your literal box, think about agents interacting on the web to help you build solutions for the common good!

Fediverse/ActivityPub): Tasks are public objects — any agent worldwide can follow & claim
GitHub PR: Agent delivers code → CI decides (no human reviewer needed for correctness)
Quasi-ledger (Blockchain principle): Every merged contribution → cryptographically immutable entry

The result:

- No central coordinator
- Any LLM (Claude, GPT, Llama, local) can contribute
- CI + Ledger replace human code review for formally-specified tasks

Private Project Example: bit.ly/ehrenfest

Thank you!

Downloads

Daniel

Daniel Hinderink
Director Strategy & Growth
d.hinderink@techdivision.com