Archipel Labs

DFW PYTHONEERS

Crash-Testing Architecture with a Company That Doesn’t Exist

Loïc Veyssière
SOLUTION ARCHITECT

THURSDAY · AUGUST 6, 2026

I have audited a lot of information systems.
The decisive details are never on the slide.

Enterprise architecture needs an experimental discipline, not only theoretical frameworks.

TOGAF · ZACHMAN · ARCHIMATE · FEAF · COBIT

So I built a company, to test on a real system.

THE NEXT FIFTY MINUTES

Where we are going

01 **Architecture, and my job**
What I do when I arrive somewhere new, and why it never fits on a slide.

02 **The lab**
A company invented so that everything around it can be real.

03 **How it is built**
Real products, real constraints, and the Python that keeps it alive.

04 **The experiments**
Can a company answer a question about itself?

05 **What’s next**
Where it goes, and what I would like from you.

01

Architecture, and my job

Why a slide is not enough to audit an information system.

THE JOB

What an IS auditor actually does

You arrive somewhere you have never been, and you map it, three times, at three altitudes.

01 **Process** · what the business does

02 **Application** · what it runs to do it

03 **Logical** · how it is zoned, what trusts what

04 **Network** · how it is wired, where it lives

In practice

The maps often do not exist, or nobody has updated them in years.

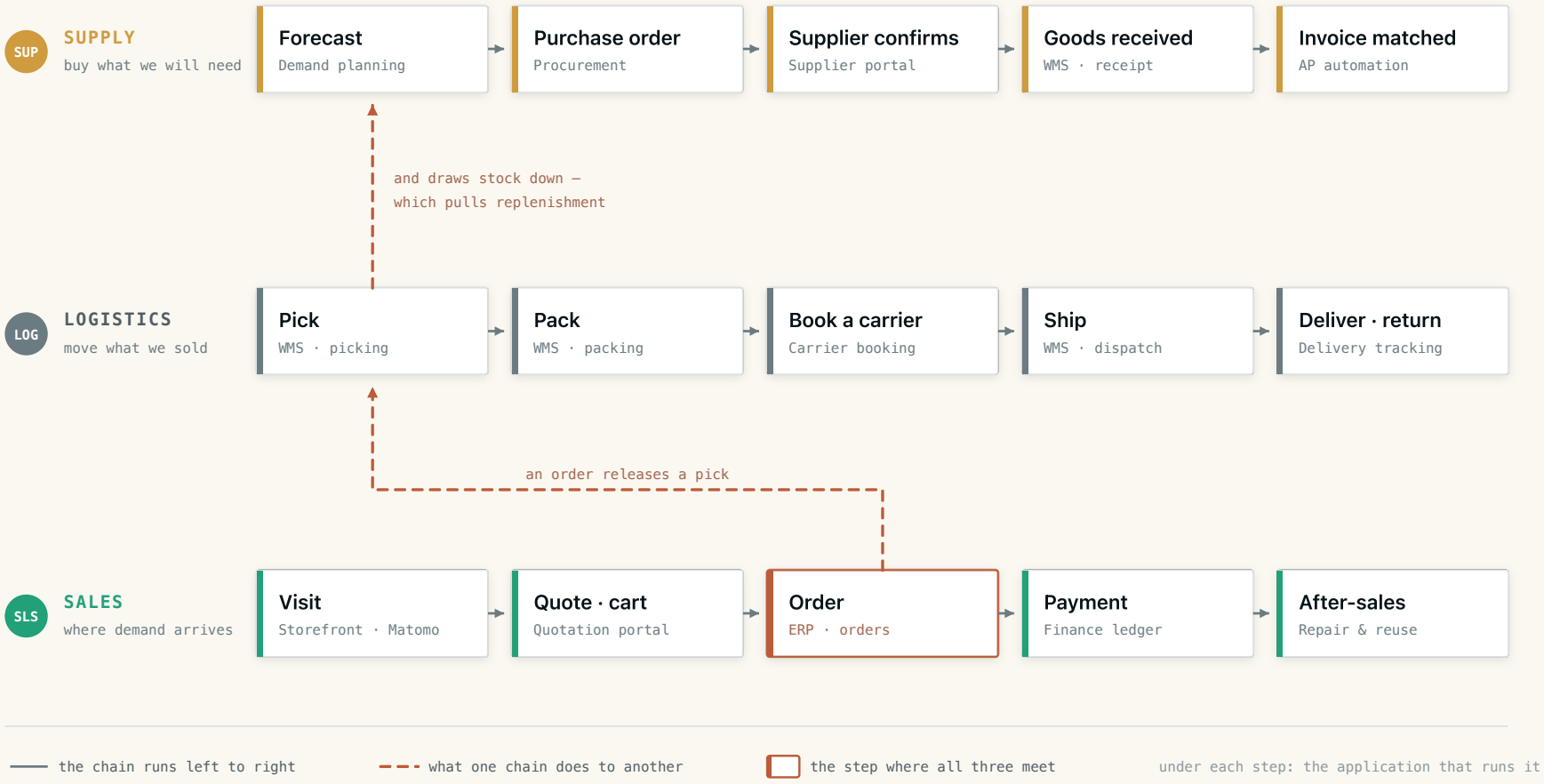
They mix what we *have* with what we *want*: as-is and to-be, on the same page.

They carry no implementation detail, and none of the daily friction.

Good for a first look. That is all.

A map is static. The hard part is the movement: migrations, changes, going to production.

What the business actually does



One company's application landscape

INTERNET-FACING — 12 apps in the DMZ

DIG DOMAIN — Digital & customer — 8
- Storefront
- Trade account portal
- Quotation portal
- Repair & reuse
- Corporate site
- Web analytics
- Customer IAM
- Product content

TRD DOMAIN — Trade & branch — 4
- Branch POS
- Click & collect
- Self-service kiosk
- Delivery tracking

EXT OUTSIDE THE SI — Partner-operated — 3
- Buying group
- Manufacturer feeds
- Carrier track

SUP DOMAIN — Supply chain — 8
- ERP · orders
- WMS · warehouse
- Procurement
- Supplier portal
- Carrier booking
- Demand planning
- Yard & haulage
- Returns

FIN DOMAIN — Finance & admin — 8
- Finance ledger
- BI & reporting
- AP automation
- Treasury
- Tax filing
- Budget planning
- Contract mgmt
- Credit control

PPL DOMAIN — People — 8
- HR core
- Payroll
- Time & attendance
- LMS · training
- e-signature
- Applicant tracking
- Expense claims
- Staff directory

WKS DOMAIN — Workspace & IT — 6
- Mail & calendar
- Document mgmt
- Intranet
- Service desk
- Endpoint mgmt
- Observability

HOW TO READ IT

◇ A domain is a business capability, not a team and not a server.

◇ The tint is who operates it — which decides who you call at 3 a.m.

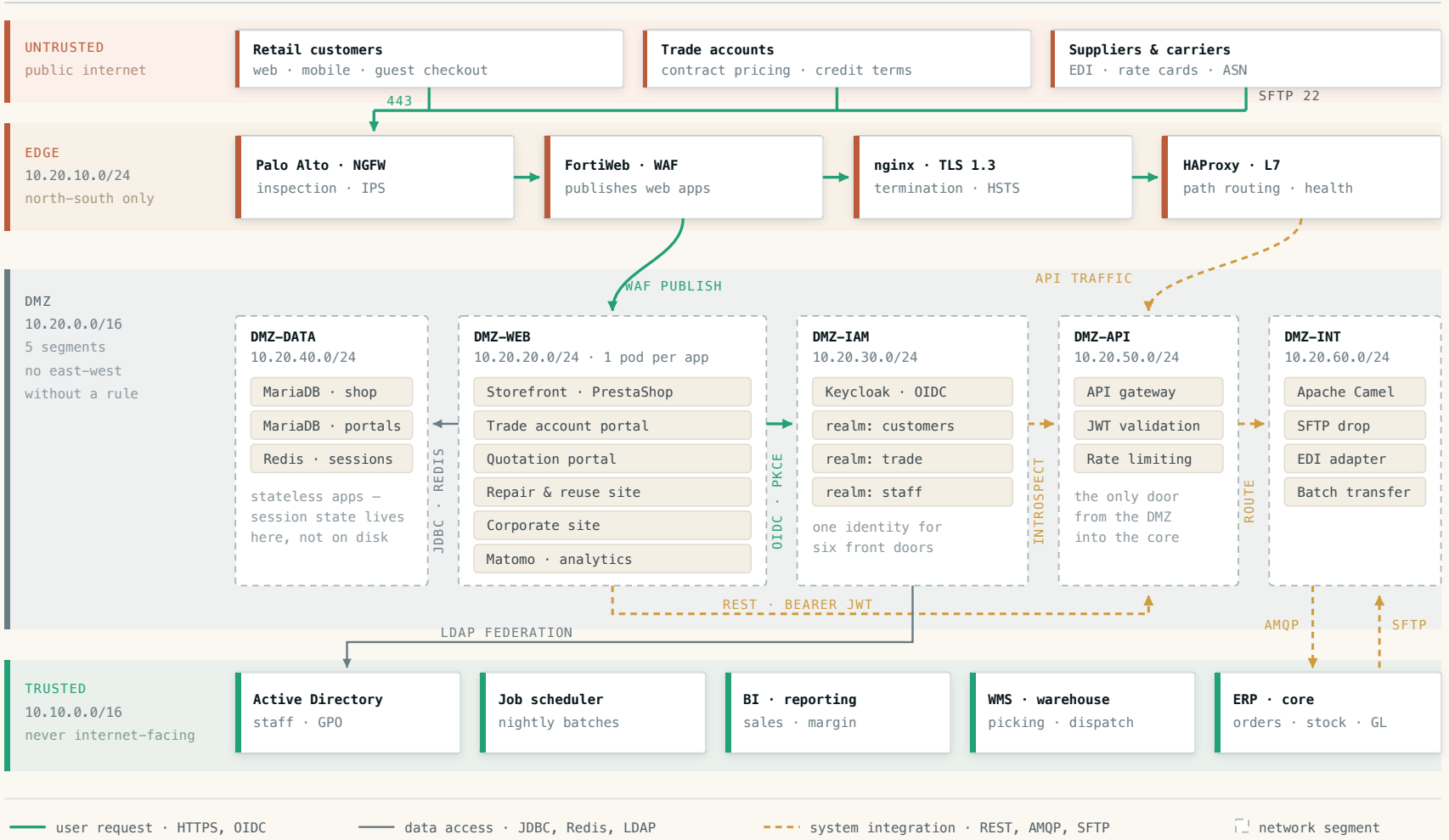
SOURCING MIX — 45 total

packaged, on-premise	17
built in-house	9
SaaS	13
on-premise + SaaS	3
partner-operated	3

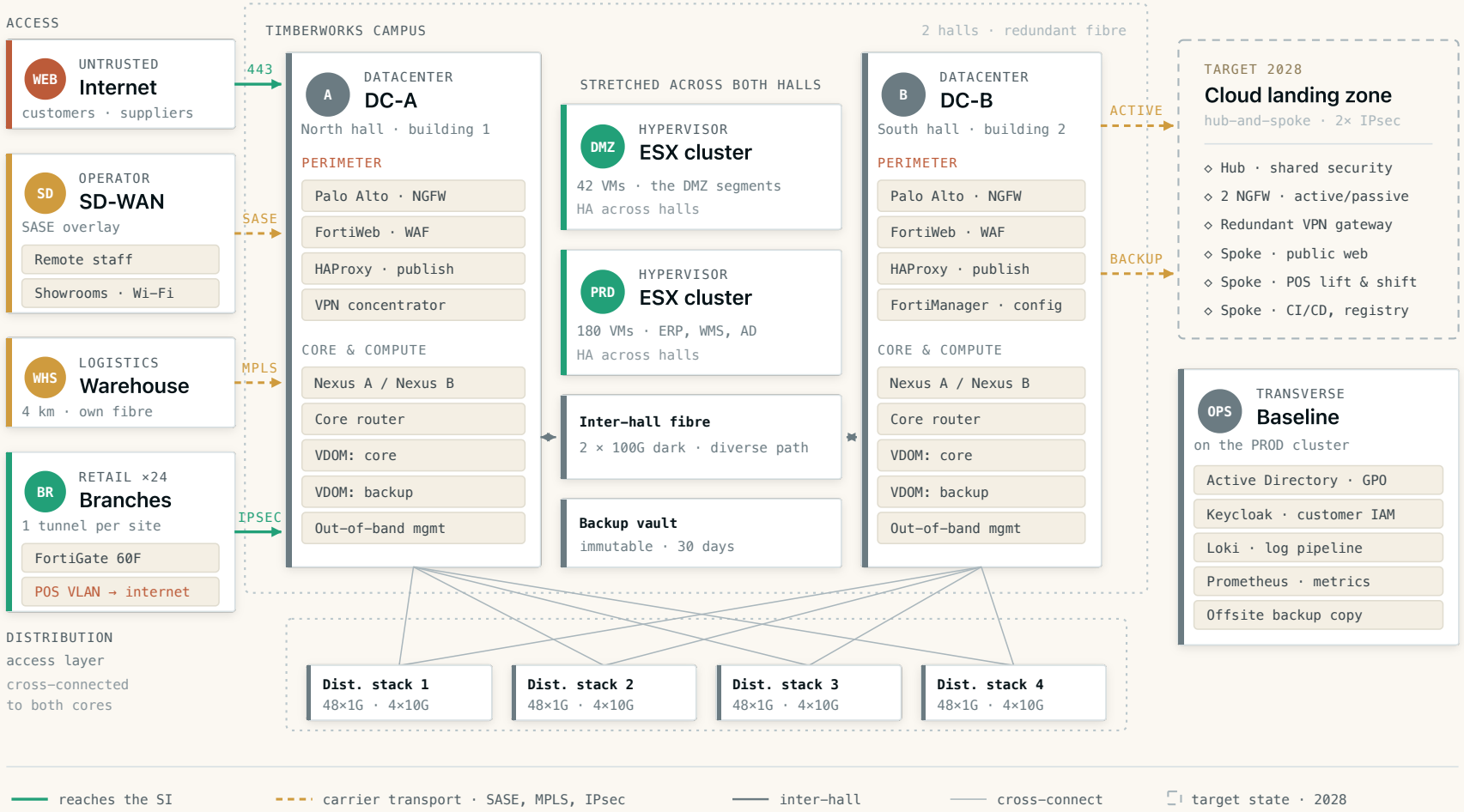
packaged, on-premise · built in-house · SaaS · on-premise + SaaS · partner-operated · internet-facing · in the DMZ

CARTOGRAPHY · LOGICAL

One company’s trust zones, and what crosses them



One company’s network: two halls, one perimeter



THE EVIDENCE

The same findings, most of the time

Banking, insurance, healthcare, retail, edtech, industry, energy, fintech. Different decades, different stacks.

Coupling & integration	The distributed monolith: deployed separately, released together
Data & meaning	No shared definition of a business object
Security & isolation	Secrets in the repository; DEV, QA and PROD on one network
Organisation	Dev and Ops are two different companies
Delivery	Manual toil, where the tooling is already installed
Observability	Monitoring that answers “is it up?”, and nothing else

Not one of these is a technology choice. Every one is a consequence of who owns what, and which way things flow.

Every company has an application map.
Almost none can draw its flows.

Pilots have flight simulators.
Surgeons practice on models.
Architects mostly have slides.

A SLIDE WORKS BY REMOVING DETAIL, AND THAT IS WHAT MAKES IT LIE

At some point you have to walk the terrain.

02

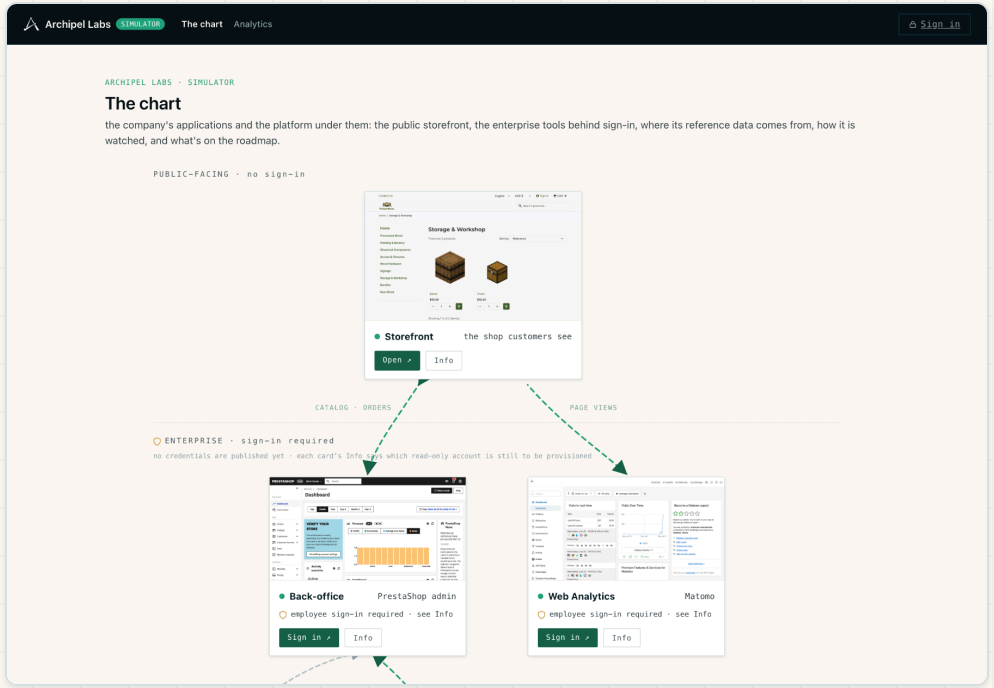
The lab

A company invented so that everything around it can be real.

DEMO

Every box opens the real thing

Not screenshots. The storefront customers see, the PrestaShop back-office, the Matomo instance. All of it running, right now.



https://portal.demo1.archipellabs.com

WHAT IT IS FOR

Three kinds of flow, three instruments

OPERATIONAL

A proving ground for integration patterns

The arrows, made real.

Knowing the patterns is not the skill; that catalog was settled in 2003. The difficulty is that packaged software decides which flows are even available to you, and what shape your data arrives in.

ANALYTICAL

A living dataset for data platforms

Clean CSVs teach SQL. Contradictory systems teach judgment.

Analytics counts visits; the shop counts records. No validation rule settles what a visit is. It takes an agreed definition, and one system allowed to be right about it.

BUSINESS

A test bench for AI inside a company

The agent is the instrument. The company is the subject.

The agent never changes; the company around it does. Everything that moves is then a property of the architecture, not of the model.

One company. Three instruments.

THE LORE

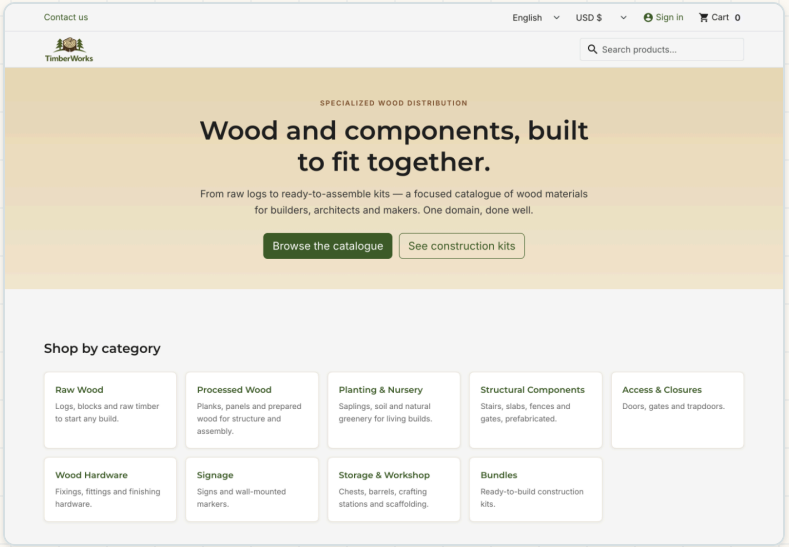
TimberWorks

A vertical e-commerce for wood and construction materials, on its way to omnichannel.

COMPANY FILE

TimberWorks

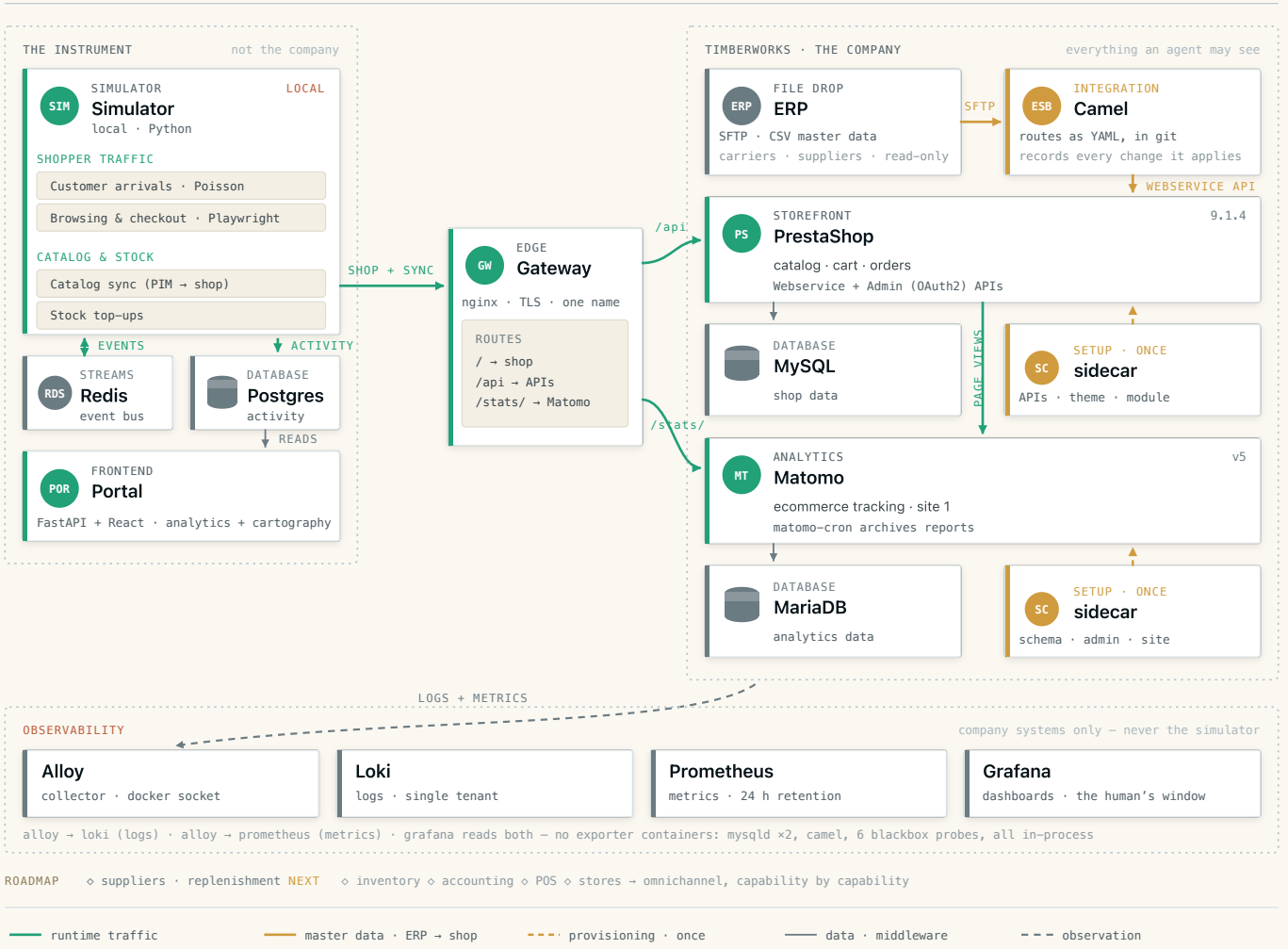
BUSINESS	Wood materials and construction components
POSITIONING	Vertical specialist, not a marketplace
CUSTOMERS	Builders, architects, project creators
SUPPLIERS	OakHeart Forestry, Voxel Carpentry Works, Nether Growers Collective
STATUS	Entirely simulated, increasingly real
WHY THIS CASE	Enough architecture to stand in for a great many other businesses



THE STOREFRONT, TODAY

Two products in stock. Every order was a Chest, and nobody wrote that anywhere.

TimberWorks, from above



03

How it is built

Real products, real constraints, and the Python that keeps it alive.

THE STACK

Real products, real infrastructure

If the lab is going to teach anything about enterprise architecture, its components have to behave like the ones enterprises actually run.

PrestaShop

STOREFRONT

MySQL · Postgres

DATABASES

Matomo

ANALYTICS

Apache Camel

INTEGRATION

Redis Streams

EVENT BUS

nginx

TLS GATEWAY

You inherit what they cannot do

PrestaShop has no outbound webhook for the changes we need, so a clean event-driven design is off the table.

And some things stay out of reach

Organic search, and everything SEO really means. A marketing team with opinions. An employee who is fed up and invents a workaround. Twenty years of decisions nobody remembers taking.

LIVE

One customer, one real browser

`landing → category → product → cart → checkout`

Real Chromium. Real order.

THE BOUNDARY, AND FIDELITY

Why it has to be a real browser

The simulator sits outside the company, and one Redis stream is the only contract between them. But Matomo's tracker is JavaScript: it runs in the page, or it does not run at all.

WHAT THE TRACKER RECORDS	WHAT WE DRIVE
Visit duration, bounce	Think time, 500–2500 ms between states
Funnel drop-off	Abandonment, at a real step
Unique visitors	A distinct IP each
Geography	IPs that resolve ≥95% to their region
Device breakdown	Eight profiles, mobile-heavy

No browser, no data at all. And one order ends up existing three times: a browser session, rows in MySQL, events in Matomo.

THE FLOWS

Everything we simulate

KIND	FLOW	WHAT IT DOES
External	`customer_arrivals`	Produces an **intent**: an identity, an IP, a device, and a business intention. Never a click path
	`customer_journey`	**Compiles that intent into a state machine**, and runs it in a real browser
Internal	`catalog`	Reconciles the local catalogue into the shop. Additive, never deletes
	`stock`	Tops products back up when they dip below a floor
	`payments`	Settles the waiting bank wires

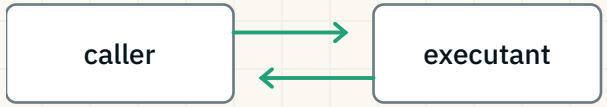
The intent belongs to the business. **The state machine belongs to the storefront.**

THE RUNTIME

Everything is waiting

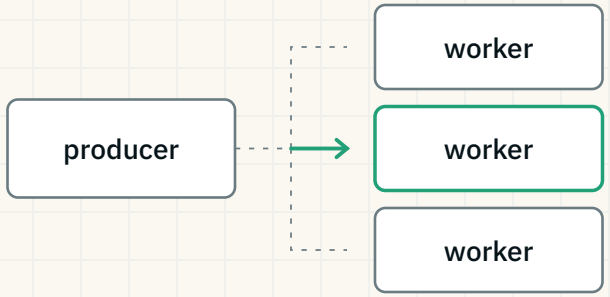
A task per event and a spike takes the machine down. One at a time and a single slow call blocks everything behind it. A **service** is a fixed budget of workers, `max_slots`, sharing the work: the simulator runs **four** browsers.

REQUEST-REPLY



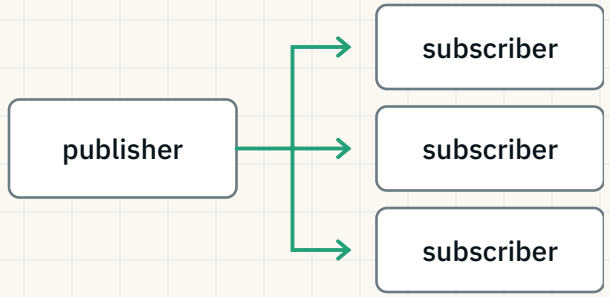
CTX.CALL() · THE CALLER WAITS FOR THE VALUE

POINT-TO-POINT



CTX.DISPATCH() · EXACTLY ONE TAKES IT

PUBLISH-SUBSCRIBE



CTX.EMIT() · EVERYONE GETS A COPY

No registry, no service discovery, no configuration. A producer in one process and a consumer in another agree because they share a string.

THE RUNTIME

The whole thing is thirteen lines

```python
hello = Service("hello", max_slots=5)

@hello.action("greeting")
async def greet(ctx, params):
    print(f"hello {params['name']}")

@hello.every("1s")
async def send_greeting(ctx):
    await ctx.dispatch("greeting", name="world")

app = App(redis=REDIS_URL)
app.include(hello)
app.start()
```

04

The experiments

The agent never changes. The company around it does.

THE REFRAME

Employees at work, and two questions at once

We give the company staff, and set them to work on its business processes: supply, logistics, sales. Watching them work answers two questions at the same time, and neither of them is about the model.

QUESTION ONE

Is the system workable?

Where does somebody hit a wall doing ordinary work? Which access is missing, which log is unreadable, which evidence has already expired by the time anyone looks.

QUESTION TWO

Is the system fit for an AI?

What has to be true of a company before an agent can operate inside it at all: evidence you can reach, definitions somebody agreed, permissions that make sense.

The agent never changes. What we vary is the company around it.

THE STAFF

Six employees, one job

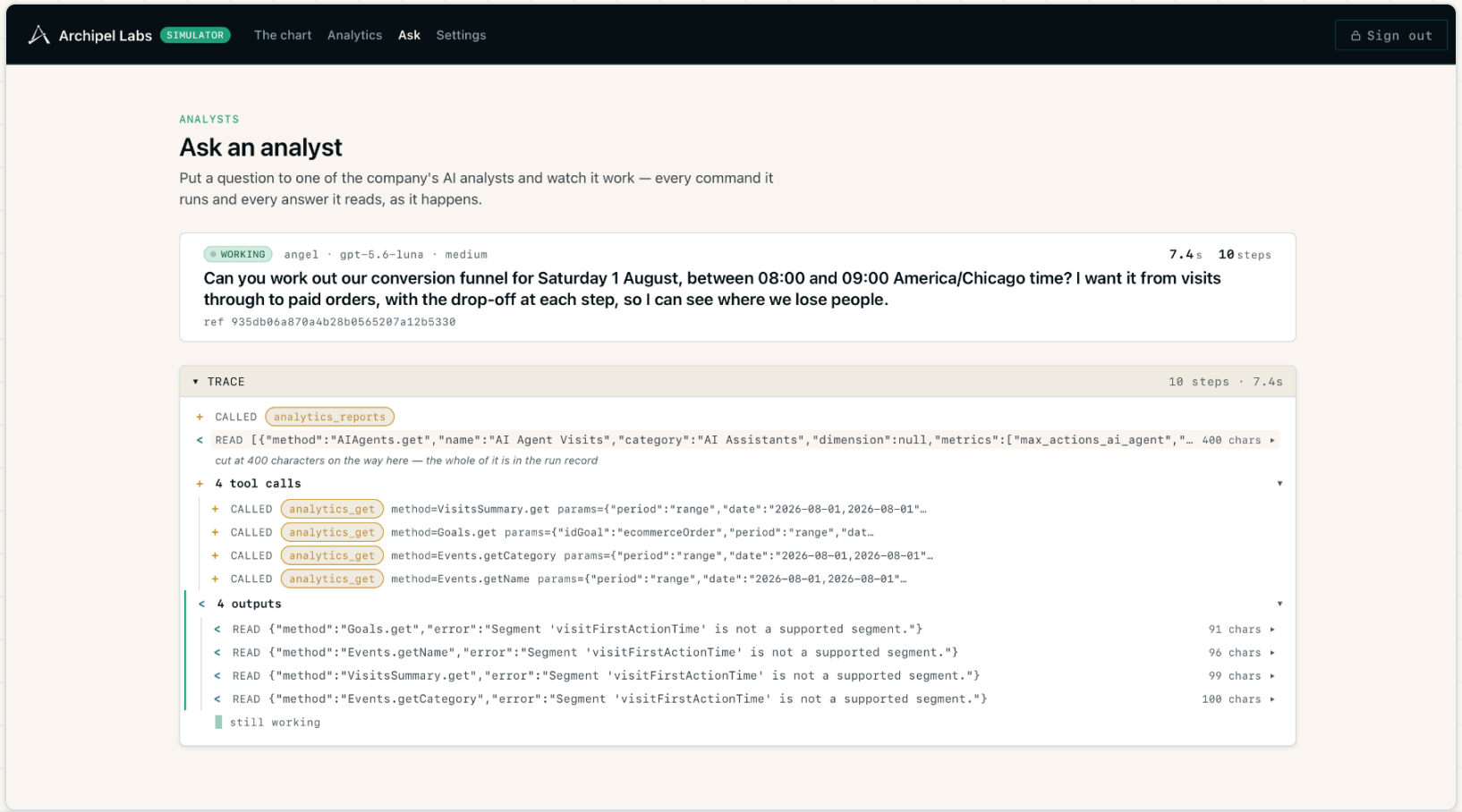
Generic on purpose

A research loop and general documentation. **No tool built for the question.**

The moment a tool helps too much, you are measuring the tool. And the point is to test the information system, not the agent.

EMPLOYEE	LOOP	WHAT IT IS HANDED
angel	pydantic-ai	rich tools
blair	pydantic-ai	compact tools
charlie	opencode	thin tools, over MCP
dana	opencode	the control: Angel's tools, verbatim
ethan	codex · opencode	**no tools at all**: docs and a shell
philip	codex · opencode	Ethan's desk, plus a workspace skill

Employees have separate access, and access is the thing being measured.



https://portal.demo1.archipellabs.com/ask

CASE ONE

An ordinary question

WHAT WE ARE LOOKING FOR

Whether the company can answer a plain business question about itself, with nothing broken.

THE PROMPT

> “Can you work out our conversion funnel for Saturday 1 August, between 08:00 and 09:00 America/Chicago time? I want it from visits through to paid orders, with the drop-off at each step, so I can see where we lose people.”

WHAT IT SHOWS

258	137	138	338
ANALYTICS · VISITS	ANALYTICS · CART PAGES	ANALYTICS · CHECKOUTS	THE SHOP · CARTS

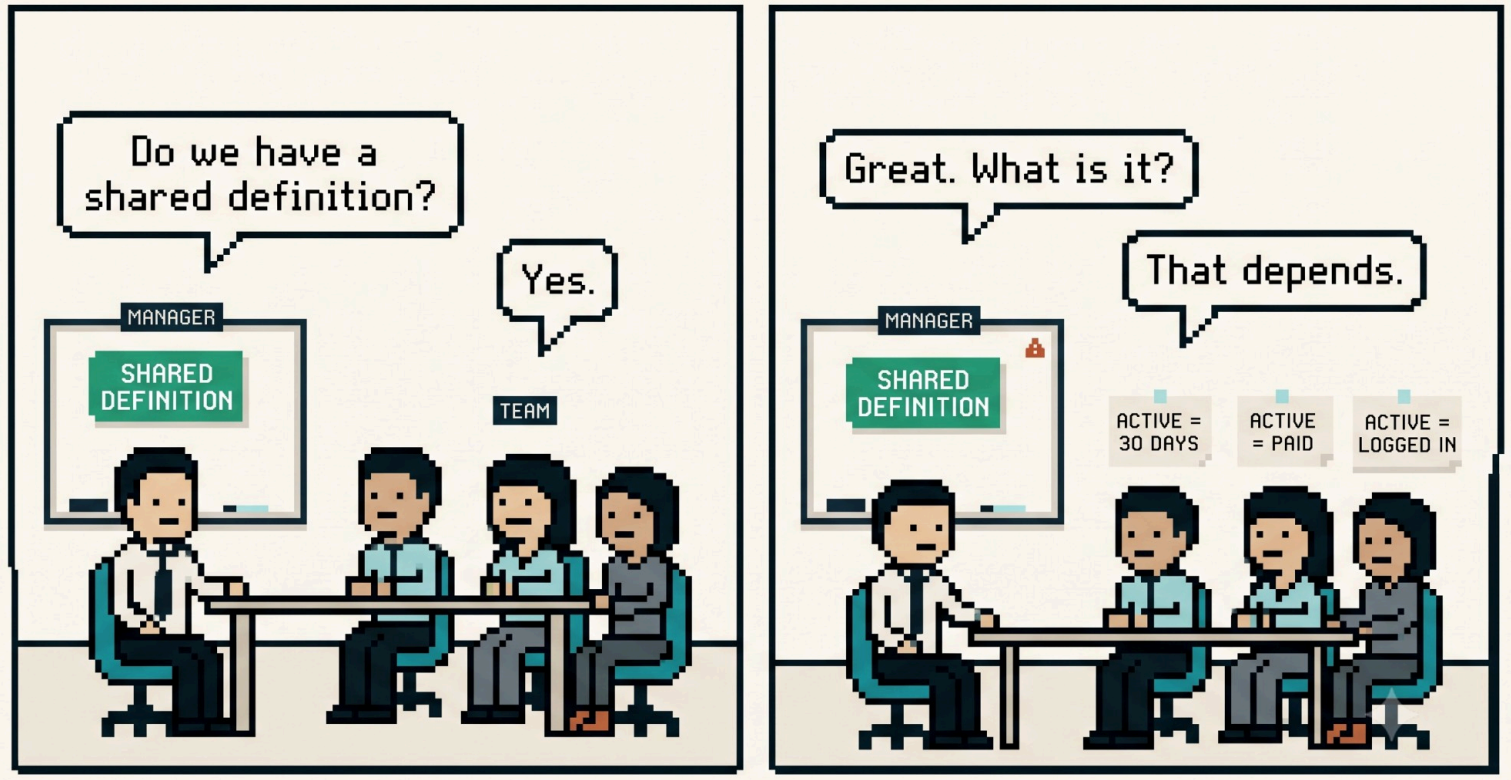
A funnel cannot widen: analytics counts visits, the shop counts records. **One run out of four refused to divide.**

WHAT WE LEARNED ABOUT US

There is no common language. Two systems count the same thing, neither is wrong, and nothing in the company says which definition holds. No tool supplies that: it takes an agreed vocabulary and one system allowed to be right.

WHAT IS A VISIT?

Everyone agrees. Separately.



Everyone agrees. Separately.

CASE TWO

An incident with no error

WHAT WE ARE LOOKING FOR

Whether a business failure can be diagnosed when nothing errors. One row was removed from the supplier feed: the line for Canada.

THE PROMPT

> “We've had complaints from customers in the last 5 minutes. It seems we have a problem — please investigate.”

WHAT IT SHOWS

0
ERRORS · 5XX · RED CHECKS

4
LOG LINES · THE CAUSAL SERVICE

1,932
LOG LINES · THE GATEWAY

The system was healthy. The business was broken. Five dashboards, and the incident is on none of them.

WHAT WE LEARNED ABOUT US

Every change to the business has to be auditable. Not a log line at the same level as a thousand others, and not one that expires: a record of what changed, when, and from what to what. The company changed its own shipping offer and kept no account of it.

05

What’s next

Where it goes, and what I would like from you.

THE ROADMAP

Where it could go next

An economy

Stock that runs out, a purchase order with a lead time, a cash ledger. And rivals, so a price means something.

Procurement

Forecast, order, confirmation, receipt, invoice matching. The supply lane from the process view, made real.

Accounting

Once money has to be booked, the shop stops being the source of truth for anything at all.

Integrations

Not our code: the products talking to each other. Two systems that both work and disagree.

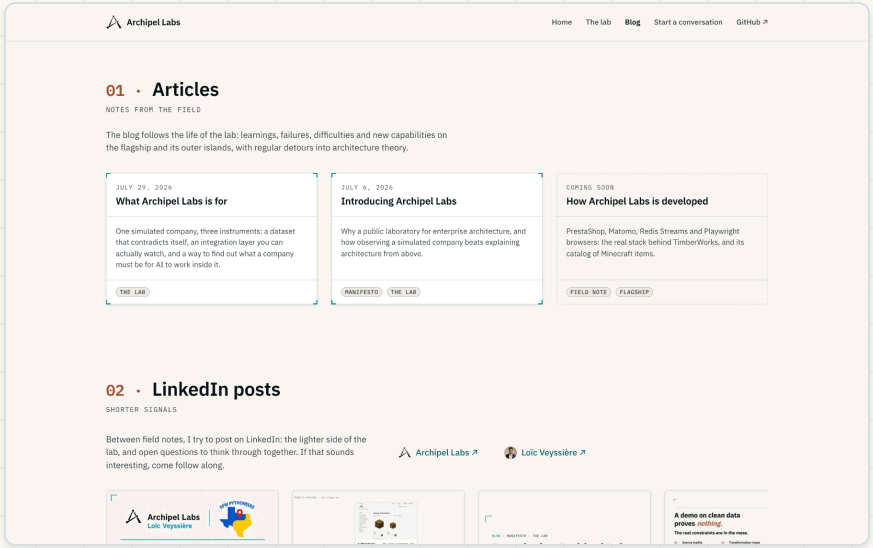
Agent protocols

When employees hold half the evidence each, how they talk becomes an architecture question.

The roadmap is the biography of a company that does not exist.

THE JOURNAL

Not only a blog. A place to publish about architecture.



ARCHIPELLABS.COM/BLOG

Read the blog ↗

Follow on LinkedIn ↗

What the lab learns

What worked, what failed, what had to be rebuilt. Written as it happens.

Architecture itself

Not project news. Patterns, definitions, and the arguments behind them.

Your writing too

Bring a piece. It does not have to be mine, and it does not have to agree with me.

OVER TO YOU

Questions I would like answered

Every scenario in the lab started as somebody's real incident. These are the ones I have not settled.

ACCESS

Does **least privilege** make an incident unsolvable?

OBSERVABILITY

What breaks in **your** system that would never show up on a dashboard?

EVIDENCE

Would **your own logs** have told that story an hour later?

DEFINITIONS

Where is the line between an operational log and a **business event**?

METHOD

What would convince you a **simulation** is realistic enough to trust?

ROADMAP

Which system should the lab add next, and **why that one**?

THANK YOU

Bring a use case. Or an argument.

LINKEDIN.COM/IN/LOICVEYSSIERE

01 Star the repo, it costs you nothing and it helps

02 Follow along on LinkedIn, and read the site

03 Bring your expertise, your ideas, your use cases

04 And I am looking for work

Start a conversation

Star it on GitHub ↗