

ANTON by openEXPERT

The ANTON family whitepapers · Autumn 2026 · Paper 7 of 11

Your Knowledge, Your Side

Models are utilities you switch. Your information, your knowledge and your way of working must stay yours — and Europe should build and use its own AI

> *Switch the model. Keep the knowledge. And let the client decide which model may read its data.*

Version 1.0 · 26 November 2026

Created by Daniel Bardun · FutureChain AB

Open source ANTON is EUPL · github.com/altspace-hub/anton_desktop

See it live ai-portal.futurechain.eu (showcases with fictional data)

Before we start

Where we left off

Last week we argued that setup beats size. With the problem, the method, the knowledge, the format and the thinking level settled first, a smaller model will often do the job — at lower cost, faster, and, if you choose, hosted in the EU. We kept the evidence honest: two small internal tests, clearly caveated, and a plain list of the tasks where a larger model still earns its price.

But that argument has a consequence, and it is the subject of this paper. **If models are this interchangeable, then the model is not the asset.** It is a utility — something you choose per task, pay for per use, and replace when something better or cheaper appears. Which raises the obvious question: *what is the asset, and where does it live?* In the language of the first paper, this is the safety gap.

The question behind the question

When compliance teams talk about AI and data, the conversation usually starts with privacy: may we send this document to that service? That question matters, and we will answer it carefully. But there is a second question underneath it that is asked far less often, and I think it matters just as much.

If we build our compliance work on AI, who ends up owning what we know?

Every organisation that uses AI seriously is teaching it something. The context it sets up, the methods it refines, the corrections it makes, the examples of what good looks like. Over months, that becomes the most valuable thing the organisation has built with AI — far more valuable than any single answer. And in many setups today, all of it sits on the provider's side, in the provider's format, under the provider's terms.

This paper is about drawing a line: the models on one side, your knowledge on the other, and a boundary in code between them that decides what may cross, to whom, and with whose permission. It is also about Europe — because the question of *where* your data is processed, and by *whom*, has better answers today than many people realise.

In this paper

- **Part 1, the problem:** the quiet lock-in, three scenes, and why "where does it go?" is so hard to answer.
- **Part 2, our thesis:** your side and the utility side; why model-agnostic is a risk control; and a fair note on where models are made.
- **Part 3, our take:** eight principles, and what we have built — with an honest status for each part.
- **Part 4, a worked example:** a client says yes to one model, and later says no.
- **Part 5:** what this means for you on Monday, and next week.

Part 1 · The problem

The quiet lock-in

Many organisations are building their use of AI the way they once built on a single software vendor. Everything sits in one provider's cloud. The prompts, the conversations, the "projects", the memory — all of it lives on the other side, in a form that only that provider's product can use.

At first this feels like convenience, and in many ways it is. The tool remembers. The team's shared workspace fills up with useful material. Each month the AI seems to know a little more about how the organisation works.

Then something changes. The price goes up. The terms are rewritten. A better model appears from a different provider. Or a supervisor asks, quite reasonably, where the client data went and who processed it. And at that moment the organisation discovers that its most valuable AI asset — everything it has taught the AI about its own work — is not really its own. It can perhaps export a pile of chat logs. It cannot take the *knowledge* with it, because the knowledge was never kept in a form that belonged to the organisation in the first place.

Nobody decided to give it away. It simply accumulated on the wrong side of the line.

Three places the data went

These are composites. No names, no clients.

Scene 1: the switching cost that was not there

A firm decides to move to a different AI provider. The new provider's models are better for its work, and cheaper. The decision is easy. The migration is not.

What does the firm keep? Chat logs, which it might be able to export. Nothing structured. Nothing the new model can actually use. Two years of carefully built context — the house style, the methods, the worked examples, the corrections — is gone. The team starts again from a blank box.

In the second of our earlier whitepapers, *"The Context Layer"*, we called this the switching-cost thought experiment. The switching cost turned out to be low, because the value kept was low. **That is not a comforting result. It means the value was never the organisation's to begin with.**

Scene 2: "where does it go?"

A compliance officer wants AI help on a client's transaction-monitoring documentation. She asks legal. Legal asks the right questions. Which company processes it? In which country? Does it keep the data after the answer is returned? Does it train on it? Does the client know?

The answers exist, but they are scattered across vendor web pages, terms of service and sub-processor lists that change without notice. Pulling them together takes weeks. The project waits — not because the answer is no, but because nobody can say, with confidence, what the answer is.

Scene 3: the client who says "only this model"

A client of an outsourced compliance desk is comfortable with one EU-hosted model and not with others. It is a reasonable position, and the client has thought about it.

The desk's tool has one switch: *AI on* or *AI off*. It cannot honour "yes to this model, no to that one". So the desk has two choices, both bad. Turn AI off for that client entirely, and lose the help. Or turn it on and hope that nobody asks which model actually read the data.

> **If you switched AI provider tomorrow, what would you keep?**

Why "where does it go?" is so hard to answer

Scene 2 deserves a closer look, because the difficulty is structural, not a matter of effort.

A single AI request can pass through several hands. Your tool sends it. A routing service may forward it. A model provider runs it, in some data centre, under some terms. Each of these is, in data protection language, a processor — and under GDPR Article 28, whoever processes personal data on your behalf needs the right contract, and you need to know who they are.

In many tools, much of this chain happens out of sight. You see the answer; you do not see which provider served it, in which country, or with what promise about retention. Worse, the chain can change between one request and the next: a busy provider is swapped for another, quietly, to keep the answer fast. For a consumer that is a feature. **For a regulated firm it means the processor chain is not a fact you can record. It is a moving target.**

And the question is no longer only about privacy. DORA, the EU's Digital Operational Resilience Act, which has applied to financial entities since 17 January 2025, requires ICT third-party risk to be managed as an integral part of ICT risk (Article 28). For ICT services that support critical or important functions, it requires exit strategies. An AI service you cannot leave without losing your work is exactly the kind of dependence those provisions are about.

Europe's question

There is a wider version of this, too, and I want to state it plainly because it is part of why we build the way we do.

For many European firms, the default answer to *"who processes our data?"* is a large provider headquartered outside the EU, under that company's terms — even when good European options exist. Some of that is habit. Some is the market's preference for size, which we discussed last week. And some is simply that nobody has checked whether the European option is good enough for the task.

That matters beyond any single firm. **A continent that only ever rents its AI from elsewhere will find it hard to set the terms on which that AI is used.** Europe now has real options: EU-hosted inference with zero data retention for strong open-weights models, and its own maker of frontier-class models. Using them where they are good enough keeps processing in the EU and builds Europe's own capacity — not as a slogan, but one task at a time.

Part 2 · Our thesis

Models are utilities. Your knowledge is not.

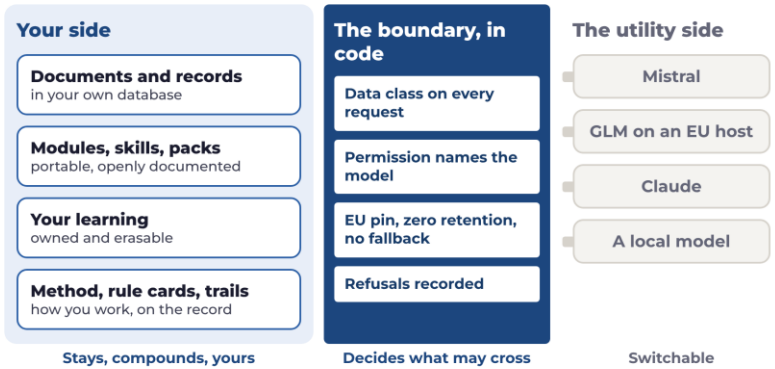
The thesis

Separate the two sides, and put a boundary between them.

On the **utility side** are the models. They are interchangeable, chosen per task, priced per use and replaceable. You should be able to switch them the way you switch electricity suppliers: with a phone call and a new contract, not a rebuild of the house.

On **your side** are your documents and records, your knowledge — the modules, skills and knowledge packs that hold your method, and the small pieces of learning each task leaves behind — your rule cards, your house style and your trails. They live in your database, in a format that is openly documented, and they compound over time.

Between the two is a **boundary in code** that decides what may cross, to which model, through which provider, and with whose permission. Not a policy document. Not a promise. Code that refuses before anything is sent, and a record that shows what happened.



Switch the model. Keep the knowledge. The boundary decides what may cross.

Why we believe it

1. Value lives in the context layer, not the model layer. This is the argument of *"The Context Layer"* and of our essay *"Beyond AGI: The Case for APCI"*. Models improve and get cheaper every few months, and the gap between the best and the fifth-best keeps narrowing. What compounds is your professional context: the method, the sources, the house view, the record of what was decided and why. **That should not be locked to any one model**, because the model is the part that will change.

2. Model-agnostic is a risk control, not a convenience. Regulated firms already treat dependence on a single supplier as a concentration risk. They ask about exit plans for their core banking system, their cloud provider, their screening vendor. An AI service deserves the same question. Being able to switch the model without losing your work is part of an exit strategy — and DORA, where it applies, expects one for services that support critical or important functions.

3. Data protection follows the processor. Under GDPR Article 28, whoever processes personal data for you is a processor and needs the right contract. That makes the choice of where and by whom a model runs a data protection decision, not a technical preference. And because clients differ, it should be made **per client, per model, on the record** — not once, for everyone, by whoever set up the tool.

4. Europe has real options now. EU-hosted inference with zero data retention exists for strong open-weights models. And Europe has its own maker of frontier-class models in Mistral, a French company, whose newest large model — Mistral Large 4, nicknamed "Le Chonk" by Mistral itself — was released in public preview in October 2026. We have built these options in where they are good enough, because keeping processing in the EU is worth doing whenever the quality allows it.

An analogy: the house and the electricity supplier

Think about how you relate to your electricity supplier. You choose one on price, on reliability, perhaps on how the power is generated. If a better offer comes along, you switch. The lights do not go out. Your house does not have to be rewired. Your furniture, your books, your family's photographs — none of it belongs to the supplier, and none of it moves when the supplier changes.

Now imagine a supplier that, over the years, had quietly come to own your furniture. Each time you plugged something in, a little more of the house became theirs. Switching would mean leaving with an empty house.

That is roughly where many organisations are heading with AI. **The fix is not to stop using electricity. It is to keep the house in your own name** — and to make sure the meter tells you exactly who supplied what.

A fair note on where models are made

There is a distinction that is easy to blur and important to keep sharp: **hosted in the EU** and **made in the EU** are two different things.

GLM is an open-weights model made by Zhipu AI, a Chinese company. Open-weights means the trained model is published, so others can host it. In the family, GLM runs on an EU inference provider with zero data retention, and Zhipu's own service is never used. That makes it EU-*hosted*. It does not make it European AI, and we will not call it that.

Mistral's models are made by a French company. On Mistral's EU endpoint, they are both made and hosted in the EU. A large US model in a US cloud is neither — unless an EU region is contracted. And a local model running on your own machine is in a category of its own: nothing leaves it at all.



Hosting and origin are two questions. Ask both.

Some organisations care only about hosting: where the data is processed, under which law. Others care about origin as well. Both positions are reasonable. **Our job is not to choose for you. It is to make both answers visible, so that you — and your client — can choose.**

What we are not claiming

We are not claiming "sovereign AI". That phrase promises more than any setup that uses hosted models can deliver. We describe exactly what runs where, and let the description speak for itself.

We are not claiming that anything makes you GDPR-compliant or DORA-compliant. The boundary helps you make and evidence the right decisions. The decisions, and the compliance, remain yours.

We are not claiming that zero data retention has been independently audited. It is the provider's commitment, as listed by the routing service we use. We pin our calls to providers that make it; we have not audited them, and we say so.

And we are not claiming that "your data never leaves". With a hosted model, the request leaves — that is how hosted models work. What we claim is narrower and more useful: **only what the task needs goes, only to the provider you allowed, and the record says so.** With a local model, nothing leaves at all.

Part 3 · Our take: your side, on the record

Eight principles

1. **Models are utilities.** Every family product can switch models. The method, the knowledge and the records do not change when the model does.
2. **Your data stays with you.** Your documents and records live in your own database, on your own machine or server. Only what a task needs goes to a model.
3. **Your knowledge is yours, in an openly documented format.** Modules, skills, knowledge packs and run records travel as .anton files, documented in ANTON's public repository. WP10 opens the format.
4. **Your learning is yours.** The small pieces of knowledge that each task leaves behind — we call them atoms — are stored in your database, owned by a person or the organisation, and can be corrected or erased. WP8 is about them.
5. **A boundary in code.** Each request declares what class of data it carries: public, synthetic or client. Code refuses a model the rules do not allow, before anything is sent. A refusal is recorded.
6. **Permission names the model.** For client data, the client's signed permission names the exact AI engines allowed. No permission, no AI.
7. **EU first, zero retention, no training.** Hosted calls are pinned to EU providers that keep nothing and collect nothing, with no fallback to other providers.
8. **The processor chain is visible.** For every call: which model, which provider, under which terms, at what cost — in "What went in", and in the client's data-handling statement.

Principle six is the one I want to dwell on, because it is the answer to Scene 3. *AI on* or *AI off* is the wrong question for a client. It treats every model as the same, when they differ in where they run, who makes them, how long the provider keeps the data and under which law. **The right question is: which models, hosted where?** And once the client has answered it, the answer has to be enforced by the system — not remembered by the team.

What we have built

Statuses matter here more than anywhere, so each part is marked.

Model-agnostic, everywhere (live and merged)

ANTON works with Claude, OpenAI, Azure OpenAI, Gemini, Mistral, local models through Ollama, and any OpenAI-compatible endpoint. The seven layers work the same on all of them, so your method does not change when the model does.

Gabriel, the Gap Assessor, can run on Azure OpenAI through a private endpoint in your own cloud tenancy, or on a local model — in which case nothing leaves your network, with no tracking and no call home. It also supports Claude, OpenAI, Gemini and Mistral. ARIS works with several providers too, and degrades gracefully when none is set: every AI feature says it is unavailable, and nothing else changes. **That last point is worth noticing. A product that still works without AI is a product that does not hold your work hostage to a model.**

Local options (merged)

ANTON is local-first: it runs on your own machine, and with a local model through Ollama, nothing leaves it. ANTON's embeddings — the step that makes your documents and memory searchable — run locally too, so even search does not need an outside service.

The EU zero-retention pin (live)

Every family showcase that uses hosted AI sends its calls through a routing service, OpenRouter, pinned to **one** named EU inference provider, with zero data retention, data collection denied, and no fallback to any other provider. "No fallback" is the important part: if that provider is unavailable, the call fails rather than quietly going somewhere else. Valentin's public health check shows the pin live, and Valentin refuses any call whose pin is not exactly right.

On the showcases, the provider for GLM is Inceptron, an inference company based in Sweden.

The data boundary (live in Valentin; merged in Olivia)

In **Valentin**, every request declares its data class, and the router refuses a mismatch before sending. Whether text counts as "public" is decided by where it came from — never by a label someone typed. Refusals are recorded. On the showcase, client data may go only to the engine allowed for client data, and since no key for that engine is installed there, **no AI runs on client data at all**. Synthetic data — invented material, declared as synthetic when the engagement is created and confirmed by a second person — and public material may use GLM 5.3 on the EU pin.

In **Olivia**, the workspace for outsourced compliance desks, client data goes to AI only when the installation *and* the client's lead have switched it on, and recorded a legal basis and a review date. Each AI request carries one client's material only, never two. And nothing that could reveal a suspicion goes to any model, at all.

Horizon never sends an institution's own records to a hosted model, and uses GLM only through an EU provider.

Client data through EU-hosted models, under signed permission (built, awaiting release)

This is the newest part, and the one that answers Scene 3 directly. It is built into Valentin and tested, but **not yet released**.

Until now, Valentin's rule for client data allowed one engine only. On 11 October 2026 we decided to widen it — carefully — so that client work can run on European options where the client says yes. The design:

- **Off by default,** per engagement and per model.
- **On only when the client's signed permission names the model** — for example, "Mistral Large 4, provider Mistral, EU endpoint, zero retention".

- **Three models may then read client data:** Mistral Large 4, offered first and pinned to Mistral's EU endpoint; and GLM 5.3 and GLM 5.3 Flash, pinned to the EU provider.
- **Four eyes on the route:** a lead proposes the model route, and a different lead or reviewer confirms it. Switching a route *off* needs only one person, because it narrows what may happen.
- **The route lapses with the permission:** a newer signed permission that names fewer engines withdraws the others automatically.
- **The database proves it:** a client call is recorded only if its route was in force when the call started, and it carried the exact pin.
- **A synthetic evaluation before first use,** and, as everywhere in the family, the AI proposes and a person decides.

Mistral Large 4: where it stands

Mistral Large 4 is Mistral's newest large model, made by a French company and released in public preview in October 2026. On 11 October we checked it live through the routing service on its zero-retention route, served by Mistral itself, and built it into Valentin, pinned to Mistral's EU endpoint. It reasons by default, so Valentin always sends an effort level and leaves room for its thinking. It is **built, not released**: not yet running in any family product.

The processor chain, made visible (built, awaiting release)

In the same design, "What went in" prints the chain for every call in plain words — for example, *"Provider: Mistral AI; zero data retention; data collection denied; no fallback to another provider"* — alongside the model, the parts sent and the cost. And the client's data-handling statement lists the processors and their GDPR Article 28 roles, computed from the record rather than written by hand.

That is the answer to legal's questions in Scene 2. Not a research project across vendor web pages, but a line in the record, for every call.

Picture: Valentin's "What went in" drawer showing the provider line (showcase, synthetic data)

Status at a glance

- **Live** (public showcases, fictional data): the EU zero-retention pin with no fallback; Valentin's data boundary; Valentin's "What went in" drawer.
- **In main code** (merged, not checked as deployed): ANTON's local options; Olivia's client-data rule. Model choice is live or merged across the family.
- **Built, not released:** Valentin's client-data routes under signed permission, including Mistral Large 4; the processor line and data-handling statement.

What actually happens when you switch

Say a better or cheaper model appears next spring, and you decide to use it for your gap analyses. **On your side, nothing moves.** The module that holds your method is the same file it was yesterday. The knowledge packs, your house style and your own documents are where they were. Your atoms stay in your database, owned by the same people. Last year's run records still name the model that produced last year's work — which is exactly what an auditor will want to see — and they remain valid evidence, because they record what actually happened.

The numbers do not move either. This is a quiet benefit of the rule we set out in the second paper: where a number has to stand up, a rule computes it, not the model. A gap score, a severity or a verdict comes from a versioned rule applied to facts a person accepted. Change the model, and the rule is the same rule; the same facts still give the same result. A system in which the model set the scores would have to re-validate every rating after a switch.

What does change is small and visible. An endpoint gets a new model on its allow-list. For client work, the client's permission must name the new model, and two people must open the route, before any client document reaches it. "What went in" prints a new model and provider line. And you test the new model on a few fictional cases first. **It should feel like changing a supplier, not moving house.**

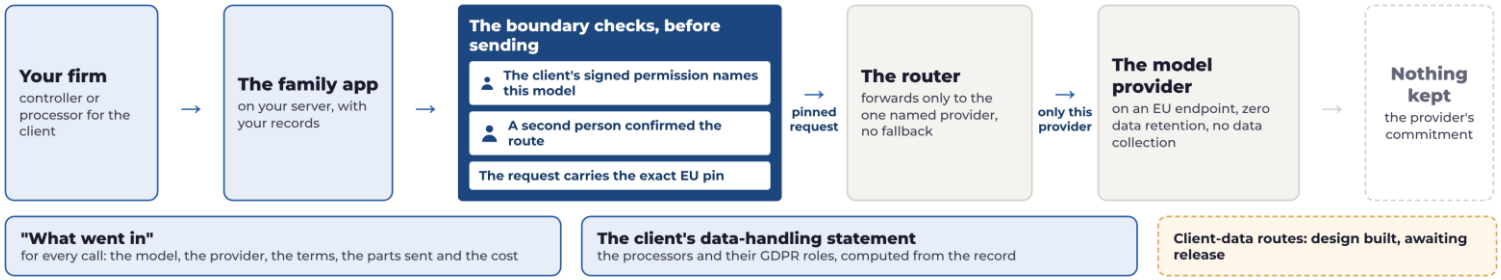
Your side and the utility side, in one table

Your side: stays, compounds, yours	The boundary, in code	The utility side: switchable
Your documents and records, in your own database	A data class on every request: public, synthetic or client	Claude, Mistral, GLM, OpenAI, Gemini, local models
Modules, skills and knowledge packs, as .anton files	Permission names the model; a route needs four eyes	Chosen per task, priced per use
Your method, rule cards and house style	EU pin, zero retention, no data collection, no fallback	Replaceable without losing your work
Your atoms: owned and erasable	Spend caps; every refusal recorded	Compared cheaply on fictional data (WP6)
Run records, trails and sign-offs	"What went in" names model, provider and cost	

Part 4 · A worked example

A client says yes to one model

The example uses Valentin and the fictional client *Exempelbolaget AB*. It follows the design described in Part 3, which is **built and tested but not yet released** — so read it as how the system is designed to behave, step by step; the status column says what is live today.



Every link in the chain is named, checked before sending, and on the record.

Step	What happens	Who acts	Status
1. AI off	The engagement starts with AI off for client data. Nothing goes to any model.	Code	Live (the data boundary)
2. Permission	The client signs a permission naming one engine: Mistral Large 4, on Mistral's EU endpoint, zero retention. Not GLM.	The client	Built, awaiting release
3. The route	A lead proposes the route for Mistral Large 4; a different reviewer confirms it. GLM has no route.	Two people	Built, awaiting release
4. A review runs	Valentin offers Mistral Large 4 first. The request carries the exact EU pin; the database checks route and pin before recording the call.	AI proposes; code checks	Built, awaiting release
5. What went in	The drawer shows the model, the provider line, the parts sent and the cost.	Code	Drawer live; provider line built
6. A second opinion	It needs a different model. GLM is not permitted, so Valentin refuses to send the client's text to it, says why, and a person reviews.	Code; a person	Built, awaiting release
7. Withdrawal	A newer permission names no engines. The route lapses by itself; the next request is refused and recorded.	The client; code	Built, awaiting release
8. The statement	The data-handling statement lists the processors, the country and their GDPR Article 28 roles, from the record.	Code	Built, awaiting release

What the example shows

Go back to Scene 3, where the desk had one switch and two bad options. In this version the client's answer — *yes to this model, no to that one* — is not a note in someone's inbox. It is a signed permission that the system reads, a route that two people opened, and a pin that every request must carry. **The client chose the model. Code kept the promise. The record shows it.**

Notice step six in particular, because it is where good intentions usually fail. Valentin's own rules want a second opinion from a different model; that is how independence is built in. A system designed only for quality would quietly reach for the next model in the list. This one refuses, because the client did not permit it, and says why. **When the client's permission and the system's own preference conflict, the permission wins.** That is what "your side" means in practice.

And notice step seven. Withdrawal is the moment most systems are weakest, because it depends on someone remembering to switch something off. Here, a newer permission naming fewer engines narrows the routes by itself. Nobody has to remember, and the lapse is on the record.

And if the client prefers a different model?

The design is not dogmatic about Europe. We offer EU options first, but the client decides. If a client permits a US-made model, under the right contracts, the same machinery applies: the permission names it, two people open the route, the pin and the processor line are recorded. **The principle is not "European models only". It is "the client's chosen models only, on the record".**

Part 5 · From here

What this means for you on Monday

1. **List what you have taught your AI tools** — the prompts, the projects, the memories, the documents — and where each one lives. Could you take it with you to another model tomorrow? If not, that is your lock-in, and it is growing every week.
2. **Map the processor chain** for your main AI use: tool, router, model provider, country, retention, training. Put it in your GDPR Article 28 records and, where DORA applies, in your DORA register of information.
3. **Ask for per-model permission.** For client work, replace "AI yes or no" with "which models, hosted where". Record the client's answer, and decide who confirms it.
4. **Test one EU-hosted option.** Take one recurring task, on fictional data, and run it on an EU-hosted model with zero data retention. Compare it with what you use now, using last week's six steps.
5. **Keep your method in your own format.** Write your modules, checklists and rule cards in a form you control — not only inside a vendor's product.

Next week: atoms

If your knowledge must stay yours, how does it grow? Next week we look at atoms: small, owned, checkable pieces of knowledge that each finished task leaves behind, so that the next task starts a little smarter, in your way of working. And we explain why memory has to earn its place — with enough evidence and enough ratings — before it is allowed to influence an answer.

A closing thought

There is a version of the next few years in which organisations rent not only their AI models but their own professional memory — and find out what that costs only when they try to leave. I do not think that is inevitable, or wise, least of all in regulated work.

The alternative is not to use less AI. It is to keep the line clear: the models on one side, chosen and switched as the work requires; your knowledge on the other, in your database, in a format you can read and carry. Between them, a boundary that does exactly what the client permitted and shows that it did.

Europe has a part to play in that, and so does every firm that chooses, task by task, to use a European option where it is good enough. **Switch the model. Keep the knowledge.** And let the people whose data it is decide who may read it.

— Daniel Bardun

See it, try it, talk to us

- **See it:** the ANTON family's public showcases, with fictional data, at ai-portal.futurechain.eu.
- **Try it:** ANTON is open source — github.com/altspace-hub/anton_desktop.
- **Read the series:** all papers in the ANTON family whitepaper series, and the earlier whitepapers, are linked from the portal. [Daniel: confirm where the PDFs will live.]
- **Talk to us:** Daniel Bardun, on LinkedIn — questions, disagreements and "this is where AI stopped working for us" are all welcome.

Important notes

Product facts were checked on 11 October 2026; statuses are summarised in "Status at a glance", and Mistral Large 4 and the client-data routes for EU-hosted models were built and tested in Valentin but not released on that date. The public showcases use fictional or public data only, and no client uses them. ANTON is open source (EUPL); the other family products are not. The products support your work and help you evidence it; they do not make anyone compliant, and nothing in this paper is legal advice.