ARIESNET ORIGINAL · EDITION 1 · 2026

The Context Supply Chain

A field guide for enterprise AI leaders

Why agents fail on context, not intelligence. The seven stages, the failure you will recognize at each, and the artifacts you keep. How to place your organization on the L0–L5 path — and how to brief the people who own the budget.

INSIDE

The seven stages of the Context Supply Chain

The three failure classes, and the stage that owns each

The L0–L5 path, and a fifteen-minute self-assessment

Ariesnet Inc

A4 edition

What is in here

Executive summary 1

Agents fail on context, not intelligence 2

The seven stages 4

Stage by stage: what breaks, what you get back 6

Where the graph comes from 9

The platform in the middle: how CoreModels sits in the chain 11

Named patterns: the rules of thumb we work by 13

The maturity path, L0 to L5 14

The business value ladder 15

The engagement ladder, mapped to the stages 17

Place yourself: a fifteen-minute self-assessment 19

How to brief your leadership 20

What an engagement looks like 21

Appendix — glossary, further reading, about 22

Two ways to read this

Every page carries a tag in the outer margin. **LEADERSHIP** pages make the argument and the business case; **ARCHITECT** pages carry the detail you would build against. A reader who has ten minutes reads only the LEADERSHIP pages and still gets the whole argument.

If you own the budget: pages 1, 15, 17 and 20. If you own the agents: read the whole thing, place yourself on the scale on page 19, and bring the placement to the interview.

A note on numbers. There are no performance percentages in this guide. The measured efficiency results we publish are ours, from our own operations; they are not a forecast of yours; anything we model for you is built from your telemetry, and a model is not a promise.

EXECUTIVE SUMMARY

Your agents are not failing because the model is weak

They are failing because three systems call the same thing three names and the tool picked one; because a superseded policy sits next to a current one with nothing to tell them apart; because the right answer was in the wiki and a near-identical page from another business unit came back instead, sounding exactly as sure.

Those are context failures. They are the agent failures we see most often in our own operations, and unlike a model they are engineerable.

The context supply chain is everything that has to be true before an agent gives you a good answer: where the content came from, what shape it is in, whether two teams mean the same thing by the same word, whether anyone checked it before the agent read it, what got selected, what the agent did with it, and what happened the time it was wrong. Seven links. A chain is worth what its weakest link is worth.

Three things this guide gives you

1. **A map.** The seven stages, each with the failure you will recognize, the artifact you are left holding, and the gate it enforces. The stages are the map; the artifacts are the work.
2. **A placement.** A six-level maturity scale (L0 Ad hoc to L5 Self-improving), applied to each stage separately. In the estates we have assessed, the failures have clustered in two stages while five were fine for now. The binding constraint is the lowest-placed stage the failing work depends on. Spending above it has not moved anything we could measure.
3. **A path.** Where the structural map of your business — the graph every serious context effort eventually needs — should come from, and why recovering it from schemas your teams already authored is cheaper and more authoritative than extracting it from documents or engineering it by hand. This is what CoreModels® does and what an Ariesnet engagement installs.

What to do with this document

If you own the budget: read pages 1, 15, 17 and 20 — this page, the value ladder, the engagement map, and the brief written to be forwarded. If you own the agents: read the whole thing, place yourself on the scale on page 19, and bring the placement to the interview.

SECTION 1

Agents fail on context, not intelligence

The industry has spent two years asking which model to use. The more useful question is what reaches the model.

An agent answering a question about your business does not know your business. It knows what it was handed: a slice of content selected by something, from somewhere, checked by someone or no one, and described by a schema or by a guess. Every one of those words hides a place where the answer can go wrong before the model has done anything at all.

The three incident classes you already count

If your team logs incidents, read them again: three classes of failure will be there.

Wrong tool	The agent had six MCP tools and picked the one whose description matched the words in the request, not the one that owned the answer.
Wrong field	The tool was right; the field it read was *status* in one system and *state* in another, and the two mean different things.
Stale definition	The definition the agent read was true in March. It was superseded in June. Nothing in the path asked.

None of these is a reasoning failure. All three are the same failure: two systems never agreed on what a thing was, and the agent inherited the disagreement.

That gives you a test you can run on a case rather than an opinion: take an incident from your own log and ask which of these three classes it lands in. If one fits, the context was wrong before the model was asked, and the fix has an address in the chain. If none fits, you are at the boundary the note below describes.

Why this is good news

A model's limits are someone else's to fix. The context supply chain is yours: your sources, your schemas, your definitions, your gates. It is the part of agent reliability that responds to engineering, and it is where the returns are if you are deciding where the next quarter's budget goes.

One honest boundary. Models do have limits, and a good assessment tells you plainly when you have hit one. This guide is about the failures that are not that.

DIAGRAM D6

The failure classes, and the stages that own them

Three incident classes

Read your own incident log again. Three classes of failure will be there, and each class is owned by particular stages.

Wrong tool

The agent had six tools and picked the one whose description matched the words in the request, not the one that owned the answer.

OWNED BY STAGE

1	2	3	4	5	6	7
Sources	Structure	Semantics	Validation	Retrieval	Agent	Learning

Wrong field

The tool was right; the field it read was called status in one system and state in another, and the two mean different things.

OWNED BY STAGE

1	2	3	4	5	6	7
Sources	Structure	Semantics	Validation	Retrieval	Agent	Learning

Stale definition

The definition the agent read was true in March. It was superseded in June. Nothing in the path asked.

OWNED BY STAGE

1	2	3	4	5	6	7
Sources	Structure	Semantics	Validation	Retrieval	Agent	Learning

None of these is a reasoning failure. All three are the same failure: two systems never agreed on what a thing was.

Each incident class is owned by particular stages. This is the diagnostic move the rest of the guide depends on: a failure is not a mood, it belongs to a link in the chain, and the link is where the fix goes.

SECTION 2

The seven stages

Sources, Structure, Semantics, Validation, Retrieval, Agent Operations, Learning Loop — and the Learning Loop returns to Sources. Each stage answers one question.

#	STAGE	THE QUESTION IT ANSWERS
1	**Sources**	What content exists, who owns it, and is it current?
2	**Structure**	Does that content have machine-legible shape?
3	**Semantics**	Does a term mean one thing across systems?
4	**Validation**	Does anything check the content before an agent reads it?
5	**Retrieval**	Does the right material actually come back?
6	**Agent Operations**	Is production behavior governed and observable — including writes?
7	**Learning Loop**	Do observed failures actually change the system?

Be sceptical of the seven-stage picture on its own

Every stage lines up with something large organizations already buy, and "our framework is proprietary" is not an argument. We use the chain because it is a good map: it tells you which conversation you are in and which stage owns a failure.

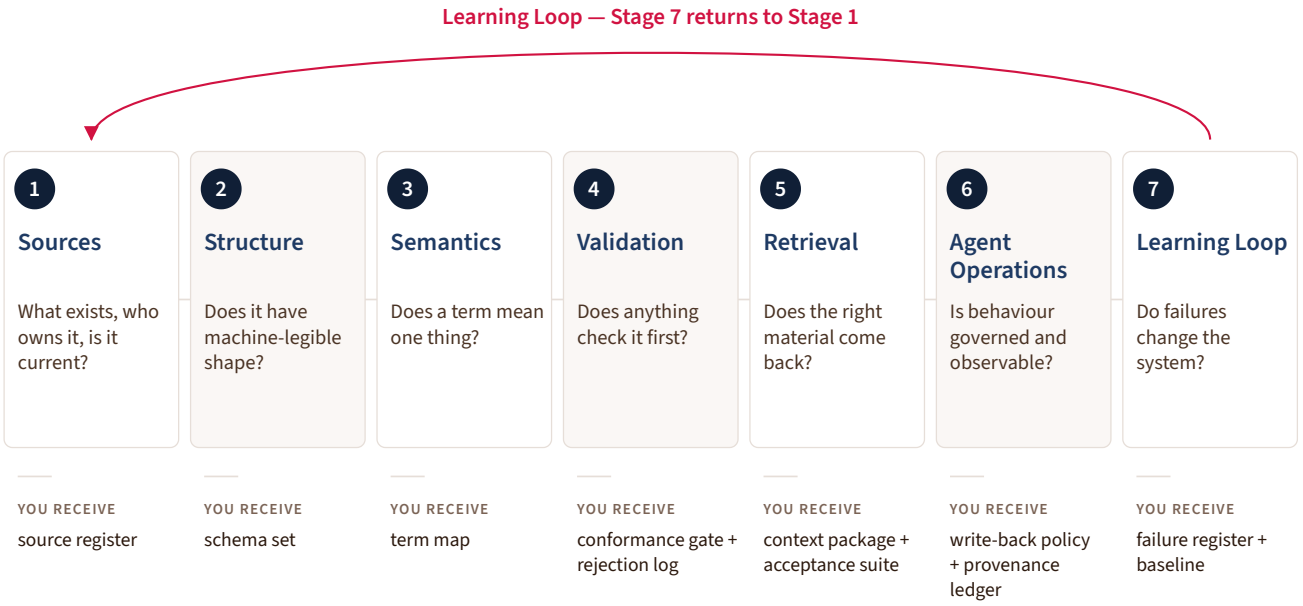
What is worth paying for is what you are holding when the work is done — a register with names on it, schemas your systems already read, a gate that refuses content and tells you why, a record of what an agent read before it answered. Those can be inspected, argued with and handed to the next team. A framework cannot.

DIAGRAM D1

Seven links, and what each one leaves you holding

The context supply chain

Seven links. A chain is worth what its weakest link is worth.



Seven links, each with the question it answers and the artifact you are left holding. The arc from Stage 7 back to Stage 1 is the part we find missing in the estates we have seen: without it, the same failure recurs for months and nothing upstream changes.

SECTION 3 · STAGE BY STAGE

What breaks, and what you get back

Each stage follows the same shape: the failure you will recognize, what the stage does about it, the artifact you receive, the gate it enforces.

1 Sources

> *Legal signed the master agreement in March. In June an agent answered a customer from a draft someone had emailed the previous autumn — confidently, with the wrong indemnity clause.*

WHAT THE STAGE DOES Every kind of content your agents touch gets one system of record and one named human owner. Copies stay copies. Where no authoritative source exists, that absence is written down instead of tolerated.

YOU RECEIVE A source register — domain, system of record, owner by name, expected change frequency, where the unofficial copies still circulate — and an agreed exceptions list.

THE GATE Content from a source nobody designated does not enter the chain.

2 Structure

> *The travel policy is a forty-page PDF. The agent could not tell a rule from an exception, or which of three effective dates applied.*

WHAT THE STAGE DOES Content is given a shape a machine can act on — typed records with fields, versions and effective dates — instead of prose every agent reinterprets from scratch.

YOU RECEIVE A published schema set in the formats your systems already read (JSON Schema, JSON-LD, ShEx) and a converted corpus of your priority content as typed records.

THE GATE Every record is checked against its shape before it counts as converted; a miss comes back with the field named.

3 Semantics

> *Sales says "enterprise." Finance says "strategic account." Support says "tier one." The agent treated them as three unrelated things, and nobody noticed until two decks disagreed in the same meeting.*

WHAT THE STAGE DOES Contested terms get one definition that wins, without asking any team to give up the word it uses. Teams keep their vocabulary; systems agree on what a customer is.

YOU RECEIVE A term map — one entry per contested concept, the definition that wins, each team's local name, the identifiers that resolve to it — and alignment to public vocabulary wherever public vocabulary exists, so private extensions cover only what is genuinely yours.

THE GATE A new term cannot arrive quietly. It matches an existing entry or is registered as new, with an owner.

SECTION 3 · STAGE BY STAGE

The gate, and what comes back

4 Validation

> *The agent cited last year's pricing policy. The document was real, in the right system, and superseded in April.*

WHAT THE STAGE DOES Content is measured against a contract — currency, ownership, conflict with other approved content, fit with policy — at the boundary, before it is available to an agent at all.

YOU RECEIVE A conformance gate with the contract stated in terms your owners can read, and a rejection log: every item that failed, what it failed on, who owns the fix. Clients use this one hardest; it is the first honest inventory of what is actually wrong.

THE GATE This stage *is* the gate. Non-conforming content does not reach retrieval — not ranked lower, not there.

5 Retrieval

> *The correct answer was in the wiki. The agent returned a near-identical page belonging to a different business unit, and sounded exactly as certain.*

WHAT THE STAGE DOES Selection on meaning, shape and entitlement rather than keyword proximity. The agent is handed the slice its task justifies and the requester is allowed to see. Discover by similarity; retrieve by structure.

YOU RECEIVE A context package definition per task — the material, each item stamped with source, version and scope — and a retrieval acceptance suite: real questions from your own operators, each with what must come back and what must not, runnable on every change.

THE GATE Material outside the requester's entitlement is never assembled. Scope is applied when the context is built, not filtered from an answer afterwards.

Stages 4 and 5 are where an estate first feels the difference. Everything before them is preparation; everything after them depends on their being true.

Acting, and learning

6 Agent Operations

> *The support assistant issued a refund outside policy. Nothing checked the limit at the moment of action, and afterwards nobody could reconstruct what the agent had read.*

WHAT THE STAGE DOES Agents run against governed context and governed tools — and are allowed to write back under an explicit policy instead of through a person copying and pasting.

YOU RECEIVE A write-back policy with its approval record, and a provenance ledger so each answer carries its receipts: which sources, which versions, which moment.

THE GATE A write outside the policy is held for a named human, not executed and reported afterwards.

7 Learning Loop

> *The same edge case failed for four months. Someone corrected it in a chat thread every time, and nothing upstream changed.*

WHAT THE STAGE DOES Failures are routed back to the stage that caused them, with an owner and a date. A correction that lives only in a conversation is an open failure, not a fix.

YOU RECEIVE A failure register classified by stage, and a telemetry baseline on the measures your own team chose.

THE GATE A failure is closed only when the change lands in the source, the schema, the term map or the policy — and the acceptance question that caught it passes on a re-run.

Seven stages, seven artifacts. If an engagement ends and you cannot point at the register, the schema set, the term map, the rejection log, the acceptance suite, the ledger and the failure register, you bought a framework.

SECTION 4

Where the graph comes from

Every serious AI context effort eventually needs a structural map: a graph of what your business objects are and how they relate. There are three places that graph can come from, and the choice decides what follows.

1. **A model can extract it from your documents.** Fast, and a statistical estimate. The graph inherits every error the extractor made, and no amount of downstream reasoning makes it authoritative afterwards. The order-of-magnitude indexing overhead attributed to "graphs" belongs specifically to this path — reconstructing by inference structure that was never captured.
2. **Specialists can engineer an ontology by hand.** Authoritative — and famously slow and expensive, and one team's reading of the business rather than the business's own.
3. **It can be recovered from structure your organization already authored.** The dbt manifest. The JSON Schemas behind your APIs. The warehouse DDL. The canonical definitions your data governance council argued about and shipped to production. Deliberate, governed, and authoritative by construction, at a fraction of the cost of either alternative.

We build on the third source. Where structure already exists, the graph is not an expense to justify; it is an asset being recovered. It is also why the conformance gate in Stage 4 can be authoritative at all: content is checked against a contract your organization authored, not against a model's guess.

If your job has been governing data assets, this is that job with one addition: the definitions your teams settled and the systems of record they live in are now the material an agent reads, so what reaches the agent is the thing to govern.

> ### Look up before you make up
>
> The cheapest reliability discipline available to an agent system: check what already exists before inventing a term, a value or a fact. In enterprise settings, a missed lookup gets called hallucination — the definition existed, and nothing pointed the agent at it.

The honest boundary again. This presupposes structure exists. For genuinely unstructured prose, vector retrieval may be entirely adequate, and a good assessment says so.

Where the graph comes from

Three places the structural map of your business can come from. The choice decides what follows.

1 Extracted from documents

A model reads your prose and infers the structure.

speed	fast
authority	a statistical estimate
cost	high indexing overhead
who authored it	the extractor

2 Engineered by specialists

An ontology team builds the model by hand.

speed	slow
authority	authoritative
cost	expensive
who authored it	one team's reading

3 Recovered from authored structure

The schemas, data models and canonical definitions your teams already fought over and shipped.

speed	deliberate
authority	authoritative by construction
cost	a fraction of either
who authored it	your own organization

We build on the third. Where structure already exists, the graph is not an expense to justify; it is an asset being recovered.

The third column is the one that changes the economics. The schemas, data models and canonical definitions your teams already fought over and shipped are the cheapest authoritative structure in the building.

SECTION 5

The platform in the middle: how CoreModels sits in the chain

CoreModels® is the platform from ARAMAI™, Ariesnet's product group, that holds the mappings and governed meaning between your schemas. It sits at Stages 2–5 and gives Stages 4–7 something authoritative to check against.

What it does, in the order a team meets it

Bring the schema your stack already produces	*dbt parse* writes a manifest; that is the whole import. Or a JSON Schema, a warehouse extract, a Salesforce object description. No passwords, no data — structure only.
Fix the meaning in a grid	Each accepted-values list becomes a governed vocabulary. The columns agents get wrong — *meaningNote*, *commonMistake*, *doNotUseFor* — are filled on the twenty fields that matter, in a grid the finance lead can edit rather than a configuration file only engineers touch. Owners are named. Changes are logged.
Align across systems	When two schemas describe the same thing, the model records that they do, and which definition wins, without rewriting either. Teams keep their vocabulary. This is Stage 3 made durable. In an Ariesnet engagement the owner of each schema agrees the mapping and your named control owner approves it; when a source schema or an element's version changes, the affected mapping reopens and the control owner accepts the regenerated render — our engagement practice, not a platform setting.
Export the contract	The governed model is exported in the formats your systems already read — JSON Schema for your services, JSON-LD for the web and graph, ShEx where that is the house standard — so evals can assert against it and pipelines can validate against it. This is what makes the Stage 4 gate authoritative rather than heuristic. When a source disconnects or a mapping goes stale, the boundary refuses rather than answers, and every refusal is logged and attributed — the conformance contract written into your engagement sets that behaviour; the platform publishes no default.
Point the agent at it	One read-only MCP endpoint, on your key. Your assistant reads the definition before it answers. Replay the question it used to get wrong.

What it is not

It is not a vector database, not a document store, not an agent framework, and not a replacement for your warehouse, your CRM or your governance council. It is the layer where those things agree on what a thing is — and the endpoint through which an agent can ask.

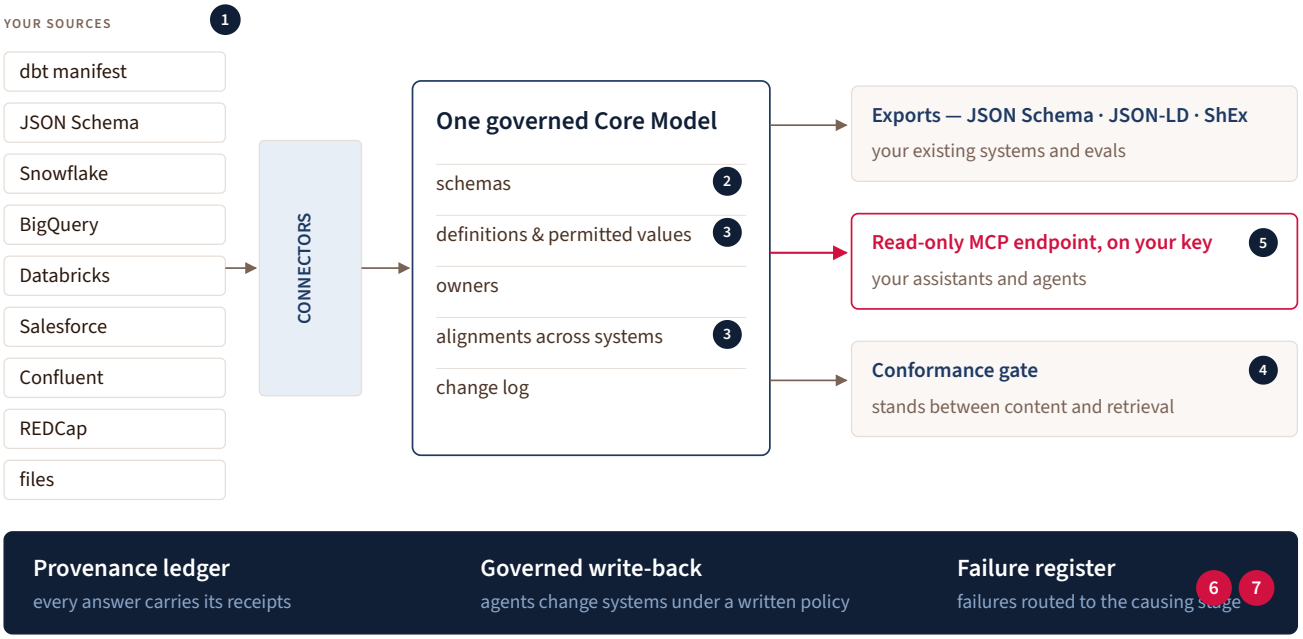
Where it is licensed. On coremodels.io (Builder, Team, Enterprise). Included in Ariesnet Implementation and Managed Operations engagements for the engagement term. Elements of CoreModels are patent pending; this guide describes what it does, not how.

DIAGRAM D2

How the pieces fit, and what never crosses the boundary

Reference architecture

Where CoreModels sits in the chain. Numerals mark the stage each part governs.



No passwords and no data cross this boundary — structure only.

Structure enters on the left and never brings data or credentials with it. The three exits are the whole product surface: the exports your systems already read, the read-only endpoint your agents ask, and the gate that stands between content and retrieval.

SECTION 6

Named patterns: the rules of thumb we work by

These have names because we reach for them enough to need them.

PATTERN	STAGE	THE RULE, STATED PLAINLY
Authoritative Source Designation	1	One system of record per kind of content, with a named owner. Everything else is a copy.
Schema Before Schema-less	2	Decide the shape first. Unstructured content joins by conversion, not by hope.
Look Up Before You Make Up	3, 5	Check what exists before inventing a term, a value or a fact.
Public Commons, Private Extension	2–3	Reuse public vocabulary where it exists. Keep private extensions for what is genuinely your advantage.
Contract at the Boundary	4, 6	Content and actions are checked where they cross a boundary, not somewhere hopeful downstream.
Scoped Context Loading	4–5	An agent receives the slice its task and entitlement justify. Nothing else travels with it.
Governed Write-Back	6	Agents may change your systems under a written policy, with approval and a record. Read-only is a stage, not a destination.
Provenance by Default	6–7	Every answer carries its receipts. By default, not on request.
Temporal Anchoring	7	Current is a question with a date attached. Content is anchored in time so it can be superseded instead of silently contradicted.
Signal Before Structure	7	Collect the signal that tells you what to fix next before building more structure on what you have.

SECTION 7

The maturity path, L0 to L5

Six levels, applied to each stage separately. Mixed placements are normal and diagnostic: a sophisticated retrieval layer sitting on ad hoc sources is a shape we have assessed more than once.

LEVEL	NAME	YOU ARE HERE IF
L0	**Ad hoc**	The answer depends on who did the work, and nobody can point to the one place a given fact lives.
L1	**Structured**	Content has consistent shape and you can query it, but two teams still mean different things by the same word.
L2	**Semantic**	Terms resolve to one definition across systems, and nothing yet checks a document before an agent reads it.
L3	**Validated**	Non-conforming content is caught before publication, but you could not say what your agents actually read yesterday.
L4	**Operational**	Production behavior is measured and attributable; fixes still depend on someone remembering to make them.
L5	**Self-improving**	A failure seen on Monday reliably changes a source, a schema or a policy — and you can show the change.

The binding constraint

The lowest-placed stage that the failing work depends on. Spending above it has not moved anything we could measure. This is why assessment comes before implementation rather than running alongside it — and why "we bought a better retrieval layer" can produce no visible change: the failures were at L0 Sources, and the new layer retrieved superseded documents faster.

SECTION 8

The business value ladder

What each step buys, in terms a finance reviewer can check.

L0 to L1 · Sources, Structure	The cost of "we didn't know that was in there" goes to a name on an exceptions list. Agent answers stop citing drafts. *Ledger:* fewer incident hours; a corpus you can count.
L1 to L2 · Semantics	Two decks stop disagreeing in the same meeting. Wrong-field incidents fall because the field now resolves to a definition. *Ledger:* reconciliation effort recovered; one term map instead of three glossaries.
L2 to L3 · Validation	Superseded content stops reaching agents at all. *Ledger:* the rejection log is the first honest inventory of what is wrong — clients tell us it is worth the engagement on its own.
L3 to L4 · Agent Operations	Agents can act, not only answer, because writes are governed and every answer has receipts. *Ledger:* the scope of automatable work widens; incident reviews take hours, not weeks.
L4 to L5 · Learning Loop	Failures change the system without a person remembering. *Ledger:* next quarter's argument is about evidence. This is where a return on knowledge assets (ROKA) statement becomes writable: what changed, what it cost to operate, what it returned, with the basis for each figure attached.

What we do not put on this page

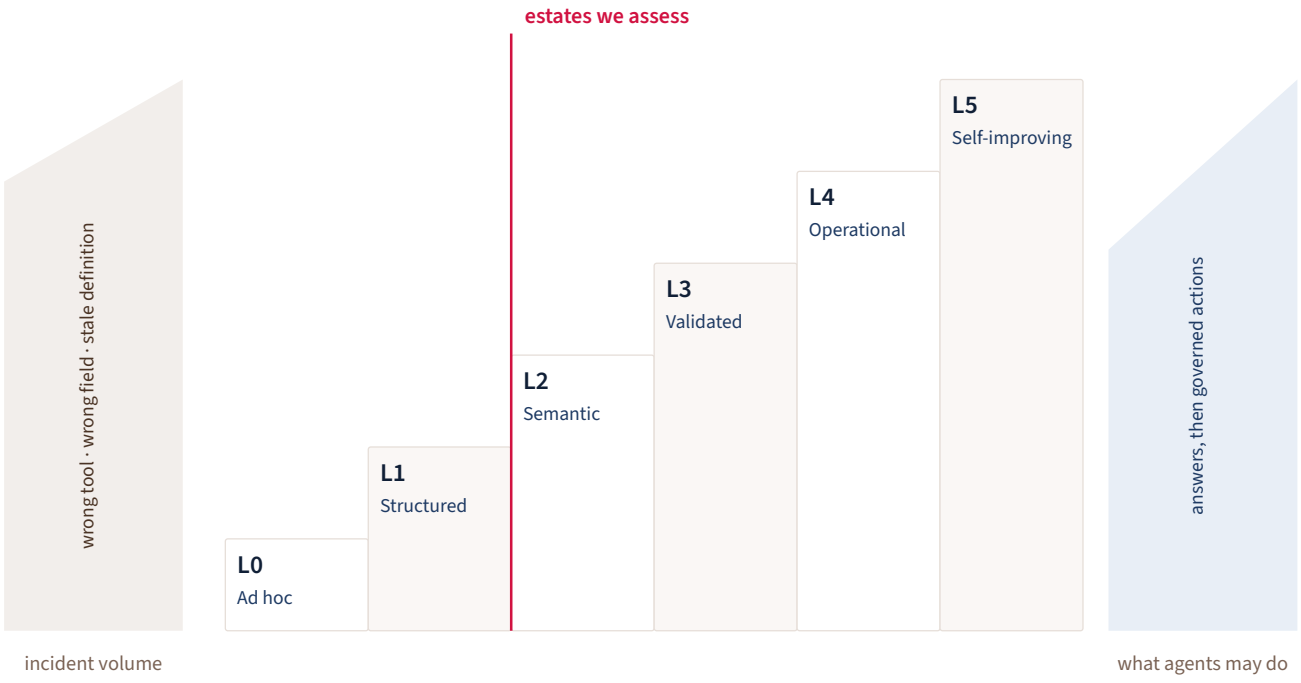
Percentages. The measured efficiency results we publish are ours, from our own operations; anything we model for you is built from your telemetry, and a model is not a promise. What we will do in an assessment is baseline the three incident classes from your own logs so the ladder has your numbers on it, not ours.

DIAGRAM D3

Where an estate starts, and what climbing buys

The business value ladder

Incident volume falls as the steps rise; what agents are allowed to do widens.



No percentages appear on this ladder. An assessment baselines the three incident classes from your own logs.

Incident volume falls as the steps rise; what agents are allowed to do widens. The mark between L1 and L2 is where the estates we assess have sat on the day they called us.

SECTION 9

The engagement ladder, mapped to the stages

Every engagement starts with an interview and every rung is fixed scope, fixed fee, priced before we start. You keep what each rung produces whether or not you climb to the next.

RUNG	ENGAGEMENT	FEE	WHAT YOU ARE HOLDING AFTERWARDS
0	**Agent interview**	30 min	A working map of your schema landscape: connectors needed, disconnects, wiring options ranked
1	**Diagnostic Briefing**	$4,500 · half day	Written findings and a buy/no-buy, within ten days of the interview
2	**Agent Context Cost & Failure Audit**	$12,000 · two weeks	Which half of the spend is buying nothing; the incident classes baselined from your logs
3	**Core Model Blueprint**	$32,500 · ~3 weeks	Stage placements with evidence; the Core Model designed; the remediation sequence
4	**Implementation**	$75,000 · first quarter	One governed model of your business behind one MCP endpoint on your key, three systems at a time; CoreModels licence for the term
5	**Managed Operations**	$9,000–18,000 / mo	The chain kept true, by your control owner, as your systems change; quarterly return on knowledge assets (ROKA) statement

We recommend starting at the two-week audit.

DIAGRAM D4

Which rung does work at which stage

Engagement to maturity

Which stages each rung does work at. Light: touched for placement. Solid: the rung concentrates here.

Every rung is fixed scope and fixed fee, priced before work starts. You keep what each rung produces.

The interview and the Diagnostic Briefing touch all seven stages lightly, because their job is placement. Every rung above them concentrates somewhere, which is what makes the fee quotable before the work starts.

SECTION 10

Place yourself: a fifteen-minute self-assessment

Seven questions, one per stage. Answer honestly; nobody else sees this page unless you bring it to the interview.

1. **Sources.** For the content your agents read day to day, can you name the system of record and the human owner without asking anyone?
   No: L0. Yes but copies circulate: L1.

2. **Structure.** Is your priority content typed — fields, versions, effective dates — or prose?
   Prose: L0–L1. Typed but unversioned: L1. Typed and versioned: L2 or above.

3. **Semantics.** Pick a contested term ("customer," "active," "enterprise"). Does it resolve to one definition your systems agree on?
   No: L1 or below. Yes, documented: L2.

4. **Validation.** Is anything refusing superseded or non-conforming content before an agent can read it?
   No: L2 or below. Yes, and it says why: L3.

5. **Retrieval.** Do you have a runnable suite of real questions with the material that must — and must not — come back?
   No: L2 or below. Yes: L3 or above.

6. **Agent Operations.** For an answer an agent gave yesterday, can you reconstruct exactly what it read? Can agents write, under policy?
   Neither: L3 or below. Both: L4.

7. **Learning Loop.** When the same failure recurs, does a source, a schema, a term map or a policy change — with an owner and a date?
   No: L4 or below. Yes, demonstrably: L5.

> **Your binding constraint is the lowest number you circled.** That stage is where the next dollar goes. An interactive version of these seven questions is at ariesnet.com/field-guide — it emails you the placement.

SECTION 11 · WRITTEN TO BE FORWARDED

How to brief your leadership

The problem in one sentence	Our agents are failing on context, not intelligence: the same thing has different names and definitions across our systems, superseded content reaches them, and nothing checks either before an answer goes out.
Why a bigger model will not fix it	The failures happen before the model runs. Upgrading the model retrieves the wrong document faster.
What we would be buying	Not a framework. Artifacts we keep: a source register, a schema set our systems already read, a term map, a conformance gate with a rejection log, a provenance record, a failure register. Plus one governed model our agents can look things up in.
Where the model comes from	Recovered from the schemas and definitions our teams already authored — not extracted by a model from documents, not hand-built from scratch. Cheaper, and authoritative because we wrote it.
What it costs to find out	A thirty-minute interview that ends in a map you keep. Then a fixed-fee ladder, priced before work starts. The Diagnostic Briefing is a half-day that ends in a buy/no-buy on rungs 2 and 3, in writing. We recommend starting at the two-week audit: two weeks on one named agent workflow, measuring cost per unit of work, and where it fails on wrong context — wrong tool, wrong field, stale definition.
What I need from you	Thirty minutes with whoever owns the agents, the list of systems our MCP tools read, and permission to baseline our incident counts.

What the interview produces in writing: the schemas you already have, the connectors your agents need, the disconnects and the remediation ranked, arriving within two business days and yours whether or not you go further. Procurement reading: "Buying context supply chain work" at ariesnet.com/resources/procurement-buying-guide — what you are buying, how it is priced, what you own afterwards.

SECTION 12

What an engagement looks like

Establishing a retrieval baseline before scaling an agent rollout

An internal support agent performed well in demonstration and poorly in production. The team assumed a model limitation and had begun scoping an upgrade. No evaluation harness existed, so nobody could say which answers were wrong or why.

We assessed the seven stages against the live estate. Sources and Structure placed at L0 and L1: the knowledge base held superseded policy documents with no supersession marking, so retrieval returned them as readily as current ones. We built an evaluation set from known-correct answers and measured the baseline before proposing any change. Kickoff to blueprint: three weeks.

> The finding was not that the model was weak. It was that no stage of the supply chain distinguished a current policy from a superseded one, so the retrieval layer had no signal to rank on.

Figures describe the engagement as delivered — scope, placements, duration. They are not projections, and the client is not named; the engagement was covered by a confidentiality term.

APPENDIX A

Glossary

Definitions as published on ariesnet.com/resources, so the guide and the site can never say different things.

Conformance gateway	The control point where context is checked against conformance and efficiency protocols before it is allowed into retrieval or agent use.
Context supply chain	The end-to-end path that sources, structures, validates, retrieves, and governs enterprise context so agents can act on material that is correct, current, and usable. The industry has a name for this work now — context engineering. The context supply chain is how we do it.
Context yield	The useful context delivered per unit of cost, latency, and operational effort—how much fit-for-purpose material the supply chain actually produces.
Governed write-back	Controlled updates from agents into systems of record, with ownership, checks, and audit so automated actions do not silently corrupt trusted data.
Grounding rate	The share of agent outputs that can be traced to approved, retrieved context rather than unsupported inference or stale material.
Learning loop	Stage seven of the supply chain: whether a failure observed in production results in a change to sources, structure, semantics or evaluation. Without an owner this stage decays silently, because nothing breaks when it stops running.
Maturity level (L0 to L5)	A six-point placement applied per stage: L0 Ad hoc, L1 Structured, L2 Semantic, L3 Validated, L4 Operational, L5 Self-improving. Stages are scored independently — a mature retrieval layer sitting on L0 sources is a common and diagnostic pattern.
Retrieval quality	Whether what comes back is the right material, established by evaluation against known-correct answers rather than by inspection. Untested retrieval can return something topically related and confidently wrong, which is the failure mode hardest to notice in production.
ROKA (return on knowledge assets)	A quarterly statement of what changed in a context supply chain, what operating it cost, and what it returned — reported with the basis for each figure attached so a finance reviewer can check the arithmetic.
Semantic layer	The shared meaning layer that aligns terms, entities, and relationships across systems so agents and people interpret the same business facts the same way.

APPENDIX B

Further reading

- Why AI agents fail on context, not intelligence
- Why MCP agents fail on schema, not reasoning — and how to map it
- Schema Landscape Map — a sample
- One governed definition, end to end
- The L0 to L5 context maturity model
- Buying context supply chain work — the procurement guide
- Look up before you make up aramai.net
- Schema authority for enterprise AI aramai.net

APPENDIX C

About

Ariesnet Inc, incorporated 1997, Texas. ARAMAI is the product group of Ariesnet, Inc. CoreModels is its platform, as part of the Schematica suite of solutions.

> ARAMAI is the product group of Ariesnet, Inc., a Texas corporation. CoreModels is its platform, as part of the Schematica suite of solutions. Ariesnet contracts, builds and integrates for clients, and operates; ARAMAI does the research and makes the software.

Contact: info@ariesnet.com · +1 214-932-3900 · ariesnet.com/interview

CoreModels® is a registered trademark. ARAMAI™ and Schematica™ are trademarks. Elements of CoreModels are patent pending. © 2026 Ariesnet Inc.