PRACTITIONER GUIDE · VERSION 2.0 · JULY 2026

Supply Chain of Intelligence

The practitioner guide: taxonomy, laws, instruments, and applications for operators and investors

ABSTRACT

This is the operational companion to the theory. It contains the full ten-layer taxonomy with all fifty sublayers, the four structural laws with their mechanisms and escapes, the three market currents, six repeatable market patterns, six company archetypes, two instruments for placing a company on the map, a defensibility audit you can run in an afternoon, worked applications for product leaders and investors, and a glossary. It assumes no prior familiarity with the framework and makes no attempt to argue for it academically; the argument lives in the companion working paper.

Anand Arivukkarasu — Kellogg MBA; former product leader, Meta (Instagram); VP / Head of Product roles at Ideas2IT, Refersion, GRIN. San Francisco. Written in a personal capacity.

Canonical source: supplychainofai.com/framework · supplychainofai.com/papers

How to use this guide

Read Part I once. Use Parts II and III as reference. Run Part VI on your own product or portfolio before you read Part VII, so the case studies land against a position you have already taken.

If you are	Start at	Then
A product leader deciding what to build next	Part VI, the defensibility audit	Part II for the layers you scored badly on, then Part VIII.1
An investor writing a thesis or running diligence	Part III, the four laws	Part V archetypes, Part VI instruments, Part VIII.2
A founder choosing where to start	Part I, then Part IV currents	Part II L1, L5, L8, then Part VII
An operator designing an agent system	Part VIII.3, the reference architecture	Part II L4, L5, L6, L8

Contents

Part I — Foundations: the definition, why supply chain rather than stack, the three registers
Part II — The map: ten layers, fifty sublayers, operator notes for each
Part III — The four laws and how to use them
Part IV — The three currents
Part V — Dynamics: six patterns, six archetypes
Part VI — Instruments: the Defensible Triangle, the Intelligence Cube, the audit
Part VII — Worked readings
Part VIII — Applications: roadmap, diligence, agent architecture, market maps
Part IX — Glossary, and how to cite

Part I — Foundations

The definition

> *Intelligence is a supply chain. Value accrues at the bottlenecks, not at the most visible node.*

That sentence names no company, no technology and no layer, which is deliberate: it is intended to survive the next five model generations. Everything else in this guide is an application of it.

Why supply chain rather than stack

A stack diagram describes how intelligence is *built*: chips at the bottom, models in the middle, applications on top. It is a technical dependency order, and it is broadly correct as one. The problem is what it implies. A stack invites the inference that value flows upward with the dependency, that the top of the diagram is where the business is, and that the layers below are inputs to be procured.

A supply chain forces different questions. Who owns the scarce input? Where is the bottleneck this quarter, and is it moving? What happens to my margin when the layer below me commoditizes the thing I charge for? Who controls the gate my output has to pass through before anyone will pay for it? Those questions have answers, and the answers change what you build.

	The AI stack	**The Supply Chain of Intelligence**
Question it answers	How is intelligence built?	Where does value accumulate, and who captures it?
Unit of analysis	Technical component	Position a competitor must pay to cross
Direction of value	Implied upward	Toward scarcity, wherever it currently sits
What it predicts	Architecture	Absorption, migration, and defensibility
Who it is for	Engineers	Operators, investors, and strategists

This is not a claim that the stack view is wrong. It is a claim that it is answering a different question, and that using it to make strategy decisions is a category error that has been expensive for a great many companies.

The three registers

The ten layers group into three registers with very different half-lives. If you remember nothing else structural, remember this table: it explains why two products that look identical to a user can have entirely different futures.

Register	Layers	Half-life	What it means for you
Substrate	L-1, L0, L1, L2, L3, L8	What users depend on. Compounds in years.	Proprietary data, trust gates, compounding memory. Slow to build, slow to lose.
Workflow	L4, L5, L6	What users live inside. Survives in months.	Sticky if deep, owned, and integrated. Absorbed if generic.
Surface	L7	What users touch. Commoditizes in weeks.	Platforms ship this for free. Placement and habit are the only durable parts.

> **The one-sentence test**
>
> Name, in one sentence, the layer you own that a competitor would have to spend years or regulatory approval to cross. If the sentence does not write itself, you do not own one yet. That is a finding, not a failure — but it should be the top of your roadmap.

What this framework does not do

- It does not tell you which company wins. It tells you which layers a company owns, which it rents, and which current is about to move value elsewhere.
- It does not replace market sizing, pricing strategy, or execution quality. A perfectly positioned company with no distribution still fails.
- It is not a maturity model. Owning more layers is not automatically better; owning a thin sliver of many contested layers is the worst position on the map.
- It is not static. The layer boundaries themselves will change; when they do, that is a versioned change, published, not a quiet edit.

Part II — The map

Ten layers, fifty sublayers. Each layer is presented the same way: what it is, the analogy that makes it stick, the five sublayers with definitions, who plays there today, the structural verdict, and operator notes — own or rent, the moves that work, the questions to ask yourself, and the characteristic failure mode.

 marks a sublayer where durable defensibility is most often observed. Company names are illustrative of a position at a point in time, not endorsements or predictions; occupancy changes far faster than structure does.

L-1 — Resources

What supports the chain. Energy, water, fabs, materials, skilled trades, the inputs the entire stack consumes.

Before chips, before data centers, before models, there's power, water, foundries, rare earths, and the humans who build it all. When AI demand explodes, the bottleneck isn't algorithms. It's megawatts and skilled trades.

THE ANALOGY

The Ground Itself, Land, Power, Materials. Before the gold rush, you need land, water rights, ore deposits, and the miners who work the seams. In AI: power generation, cooling water, foundry capacity, rare earths, and the electricians and technicians who physically build the boom. When demand spikes, this layer is the real bottleneck.

THE FIVE SUBLAYERS

Sublayer	What it covers
L-1a Energy & Grid Interconnect	Power generation, PPAs, transmission, and the multi-year grid-interconnect queue, the megawatts the stack consumes and the wait to get them switched on
L-1b Thermal & Water Management	Cooling systems, water access, immersion/liquid cooling, heat reuse, the thermodynamic ceiling on every GPU cluster
L-1c Fabrication & Foundry	Leading-edge chip fabrication capacity, EUV lithography, advanced packaging (CoWoS), the physical floor of L0
L-1d Critical Materials & Supply Chain	Rare earths, lithium, cobalt, gallium, specialty substrates, and the refining, logistics, and geopolitical chokepoints that gate them
L-1e Skilled Trades & Human Capital	Electricians, HVAC techs, data-center builders, fab process engineers, robotics technicians, the labor pool no model can synthesize

Who plays here today	Structural verdict
NextEra, TSMC fabs, MP Materials, Vistra, Bechtel	The real bottleneck. Slow to build, impossible to fake.

OPERATOR NOTES

Own or rent? Effectively unownable by software companies. Read it as a constraint, not a market.

MOVES THAT WORK

- Price your compute assuming a 2–5 year interconnect queue, not a spot GPU market.
- Treat power-adjacent geography (Texas, Nordics, Gulf) as a real input to product latency and cost.
- If your roadmap assumes inference gets 10x cheaper on schedule, write down which L-1 constraint has to break for that to be true.

QUESTIONS TO ASK YOURSELF

1. Does any part of my unit economics assume compute prices fall faster than grid capacity grows?
2. Who upstream of my vendor is physically constrained, and what is their lead time?
3. If a single fab or a single utility region slipped 18 months, which of my plans die?

> **Characteristic failure mode**
>
> Building a margin model on the assumption that the physical world scales like software.

L0 — Infrastructure

The shovels. Chips, data centers, networking, cloud, edge, what is needed to process intelligence.

The compute substrate. NVIDIA doesn't care which model wins, they sell to all of them. When L2 commoditizes, value accrues to L0.

THE ANALOGY

The Shovels & Mining Equipment. Before anyone finds gold, someone has to build the pickaxes, drill rigs, and mine shafts. In AI: NVIDIA builds the GPUs, CoreWeave builds the data centers, hyperscalers run the clouds. No shovels → no gold rush. Shovel sellers outlast most miners.

THE FIVE SUBLAYERS

Sublayer	What it covers
L0a Silicon & Memory	GPUs, TPUs, custom AI accelerators, plus HBM and high-bandwidth memory (SK Hynix, Micron, Samsung), the invisible bottleneck behind every chip cycle
L0b Data Centers	Physical facilities housing compute at scale
L0c Interconnect Fabric	Networking between chips, racks, regions, clouds
L0d Compute & State Infrastructure	On-demand compute, scheduling, and durable agent state (checkpointing, workflow state, runtime memory stores)
L0e Edge & On-Device Compute	Local inference on phones, vehicles, sensors, endpoints

Who plays here today	Structural verdict
NVIDIA, AMD, TSMC, CoreWeave, Equinix	Shovel sellers win every gold rush.

OPERATOR NOTES

Own or rent? Rent. Almost no application company should own silicon or facilities.

MOVES THAT WORK

- Negotiate for portability (multi-cloud weights, open inference formats) rather than for price.
- Instrument cost per successful outcome, not cost per token; the second number lies as models change.
- Keep one credible fallback provider warm at all times. It is your only pricing leverage.

QUESTIONS TO ASK YOURSELF

1. What percentage of my COGS is a single vendor's list price?
2. How many days would a full provider migration take today?
3. Does my product degrade gracefully if inference cost triples for a quarter?

> **Characteristic failure mode**
> Confusing a favourable startup credit programme with a durable cost structure.

L1 — Data

The raw input. What data do you have that nobody else can get?

Proprietary datasets are the raw fuel. More agents = more demand for data. The L1b test: if your data is public, the model layer wins.

THE ANALOGY

The Raw Gold Ore. The unrefined material pulled from the earth. Some mines have pure veins (proprietary data), others have common dirt (public data). Public data is already mined by everyone. The L1b test: if your data is public, the model layer wins.

THE FIVE SUBLAYERS

Sublayer	What it covers
L1a Public & Open Data	Common Crawl, Wikipedia, government data, open datasets
L1b Proprietary Data □	Licensed, paywalled, or internally generated training corpora
L1c Behavioral & Sensor Data □	Clicks, sessions, interaction logs, and camera, LiDAR, IMU, telemetry, and physical-world sensor streams for robotics and autonomy
L1d Outcome Data □	Labels, results, conversions, win/loss, audit trails, what actually happened after the model acted
L1e Synthetic & Simulation Data	Machine-generated corpora and simulated environments (Isaac Sim, CARLA, Omniverse, world-sim) for training, augmentation, and embodied agent rollout

Who plays here today	Structural verdict
Apollo.io, Bloomberg, ZoomInfo, Scale AI	Structurally safe. API-first wins.

OPERATOR NOTES

Own or rent? Own L1b, L1c, L1d. This is the most reliably defensible position available to an application company.

MOVES THAT WORK

- Design the product so that ordinary usage produces outcome data (L1d) nobody else observes.
- Write data rights into the contract on day one; retrofitting consent is close to impossible.
- Separate the corpus from the model. Corpora survive model generations; fine-tunes do not.

QUESTIONS TO ASK YOURSELF

1. What do I know about my customers' work that a frontier lab structurally cannot observe?
2. If a competitor copied my UI exactly tomorrow, what would still take them three years to accumulate?
3. Am I capturing what happened after the model acted, or only what the model produced?

> **Characteristic failure mode**
>
> Calling a scraped public dataset “proprietary” because it is stored in your database.

L2 — Models

Intelligence refinement. Rent early, build custom at scale.

Foundation models are the smelters, expensive, few can operate at scale. Once refined, the gold is a commodity, which is why model providers need to move up the chain.

THE ANALOGY

The Smelter & Refinery. Raw ore becomes pure gold through smelting. In AI: foundation, specialized, and reasoning models refine raw data into intelligence. Refining is expensive and only a few can do it at scale, but once refined, the gold is a commodity.

THE FIVE SUBLAYERS

Sublayer	What it covers
L2a Foundation & Multimodal Models	Large pre-trained generalists, GPT, Claude, Gemini, Llama, and vision-language-action and video models (Sora, Veo) that span text, image, audio, and motion
L2b Specialized & Fine-Tuned Models	Domain-tuned, distilled, and PEFT/LoRA-adapted models for specific verticals or tasks (BloombergGPT, Med-PaLM, Codestral)
L2c Embedding & Retrieval	Vector representations, search indices, reranking, and RAG infrastructure
L2d Model Routing & Composition	Selecting, chaining, ensembling, or mixture-of-experts routing across multiple models per task to balance cost, latency, and quality
L2e Reasoning & World Models	Extended chain-of-thought, planning, and multi-step inference, plus predictive world models (V-JEPA, Genie, Sora-as-simulator) that let agents and robots imagine outcomes before acting

Who plays here today	Structural verdict
OpenAI, Anthropic, Google DeepMind, Meta AI	Winner-take-most. Commodity risk high.

OPERATOR NOTES

Own or rent? Rent early. Fine-tune only where a measurable domain gap survives the next frontier release.

MOVES THAT WORK

- Abstract the model behind an internal interface; assume you will swap it twice a year.
- Route by task economics (L2d): cheap model for volume, frontier model for the 5% that carries risk.
- Keep an evaluation set that is about your customers, not about benchmarks.

QUESTIONS TO ASK YOURSELF

1. If the frontier model improves 30% next quarter, does my product get better or get redundant?
2. What do I do that survives the model owner shipping the same loop for free?
3. Is my fine-tune a moat, or a maintenance liability?

> **Characteristic failure mode**
>
> Treating a system prompt as intellectual property.

L3 — Gatekeeping

Trust, acceptance, approval. Can the system be allowed in?

Without the hallmark, no enterprise buyer touches it. L3 is the slowest moat to build and the hardest to replicate.

THE ANALOGY

The Hallmark & Assay Office. Before gold enters the market, the assay office verifies purity and the hallmark guarantees quality. In AI: compliance, evals, safety, editorial taste, and distribution control are the gates. Without the hallmark, no enterprise, and no app store, lets you in.

THE FIVE SUBLAYERS

Sublayer	What it covers
L3a Compliance & Export Controls	Regulatory, legal, and policy filters (HIPAA, GDPR, SOC 2, EU AI Act), plus chip export controls, model sovereignty, and data-residency regimes that decide where the stack is allowed to run
L3b Quality Gates	Accuracy, hallucination detection, output grading, eval harnesses, regression suites
L3c Safety, Security & Provenance	Harmful-content filtering, adversarial defense, prompt-injection protection, and content provenance (C2PA, watermarking, deepfake attestation) that proves what was generated and by whom
L3d Editorial Gates □	Tone, brand voice, style, taste, the human judgment layer
L3e Distribution Gates □	App store approval, ranking, marketplace curation, discovery control

Who plays here today	Structural verdict
Vanta, Drata, OneTrust, Apple App Store	Essential. More agents = more access control.

OPERATOR NOTES

Own or rent? Own it if trust is your product. Otherwise integrate with the incumbent verifier rather than competing with it.

MOVES THAT WORK

- Map the certifications your buyer must hold, then decide whether you help them hold it or hold it for them.
- If you generate, do not also claim to verify your own output; buyers discount self-attestation to zero.
- Build the evidence trail (who approved what, when, on what basis) as a first-class product object.

QUESTIONS TO ASK YOURSELF

1. Whose signature is on the line when my output is wrong?
2. Would my buyer's auditor accept my own logs as evidence?
3. Is there a regulator, insurer, or board committee that already requires a second party here?

> **Characteristic failure mode**
>
> Assuming a capable model removes the need for an independent verifier. It never has, in any industry.

L4 — Access

Connectivity, permissions, integrations, the pipes layer.

Grammarly survived because it had railroad tracks (plugins) into Word, Gmail, Chrome. Jasper had no tracks. Deep integrations and agent identity create switching costs.

THE ANALOGY

The Railroads & Transport. Refined gold needs to move, by rail, armored truck, secure vault. In AI: APIs, MCP, real-time pipes, and agent identity move intelligence between systems. Grammarly survived because it had tracks into every workflow. Jasper had none.

THE FIVE SUBLAYERS

Sublayer	What it covers
L4a API & Integration Layer	REST/GraphQL endpoints, SDKs, webhooks connecting AI to systems
L4b Agent Interface Protocols □	MCP, tool-use specs, agent-to-agent communication standards
L4c Access Governance & Agent Commerce	Who can use what, RBAC, scoping, audit trails, and agent-payment rails (Stripe/Visa/Mastercard agent-pay, spend limits, programmatic checkout, machine-to-machine billing)
L4d Real-Time Interaction Infrastructure	Streaming, voice pipelines, video, low-latency modality transport
L4e Agent Identity & Provenance □	Verifying which agent acted, credential chains, trust signatures

Who plays here today	Structural verdict
AWS, Snowflake, Supabase, Twilio	Load-bearing walls. Invest accordingly.

OPERATOR NOTES

Own or rent? Own your integration surface. Rent the protocols.

MOVES THAT WORK

- Write back into the system of record; read-only integrations are trivially replaced.
- Adopt open agent protocols early but keep the permission and audit model proprietary.
- Treat agent identity (L4e) as a product feature the moment an agent acts on a customer's behalf.

QUESTIONS TO ASK YOURSELF

1. If my integration disappeared overnight, what would break in my customer's workflow?
2. Do I own the credentials and the audit trail, or does my customer's IT department?
3. Is any integration of mine a thin wrapper on a public API that the platform could publish itself?

> **Characteristic failure mode**
>
> Counting logos on an integrations page as if they were switching costs.

L5 — Execution

Applied skills and capabilities. Doing the actual work.

A jeweler takes refined gold and crafts rings, necklaces, watches, each requiring specialized skill. L5 is THE entry point for AI-native companies.

THE ANALOGY

The Master Jeweler. A jeweler takes refined gold and crafts rings, necklaces, watches, each requiring specialized skill. In AI: domain skills, decision frameworks, and operating playbooks transform generic intelligence into specific capability. Harvey knows legal. Sierra knows CX.

THE FIVE SUBLAYERS

Sublayer	What it covers
L5a Domain Execution & Tool Use	Doing the actual work, legal drafting, code generation, diagnosis, underwriting, including function calling, code interpreter, browser/computer use, and structured tool invocation that turns a model into an operator
L5b Decision Frameworks & Reasoning Scaffolds	Structured thinking patterns, checklists, rubrics the agent follows
L5c Retrieval-Augmented Workflows	Grounding execution in retrieved context, knowledge, and documents
L5d Operating Playbooks	Company-specific SOPs, rules, preferences encoded for agents
L5e Interaction Skills & Actuation	Tone, empathy, negotiation, persuasion, and physical-world actuation (robotic control, valve/vehicle/device operation)

Who plays here today	Structural verdict
Harvey, Sierra, 11x, Cursor	Durable if deep. Generic skills get absorbed.

OPERATOR NOTES

Own or rent? Own it, deeply and narrowly. Generic execution is absorbed; encoded domain judgement is not.

MOVES THAT WORK

- Encode the decisions experts make, not just the tasks they perform.
- Ship playbooks (L5d) that customers configure and then depend on; configuration becomes lock-in.
- Measure work completed to an accepted standard, not outputs generated.

QUESTIONS TO ASK YOURSELF

1. Could a competent generalist with a frontier model reproduce 80% of my value in a weekend?
2. What judgement in my domain is expensive to be wrong about, and do I encode it?
3. Do my customers hire fewer people because of me, or just produce more drafts?

> **Characteristic failure mode**
>
> Depth theatre: many features, none of them load-bearing in a real workflow.

L6 — Orchestration

Workflow, routing, coordination. How skills compose into outcomes.

A single ring is useful. A curated jewelry collection with tasting, fitting, and custom design is an experience. Orchestration composes skills into workflows.

THE ANALOGY

The Jewelry Store & Workshop. A single ring is useful. A curated collection with fitting and custom design is an experience. In AI: orchestration composes individual skills into multi-step workflows with human override and runtime assurance. One skill → one task. Orchestration → entire workflows.

THE FIVE SUBLAYERS

Sublayer	What it covers
L6a Agent Loops	Single-agent plan-act-observe cycles
L6b Human-in-the-Loop □	Escalation patterns, approval workflows, human override design
L6c Role Routing & Task Decomposition	Breaking complex work into subtasks and assigning to the right agent
L6d Context & State Management	Maintaining working memory, session state, context windows across steps
L6e Runtime Assurance & Learning Loops	Post-deployment monitoring, evals, feedback pipelines, drift detection

Who plays here today	Structural verdict
LangChain, CrewAI, Zapier (at risk), Make (at risk)	Contested. Becoming a feature, not a product.

OPERATOR NOTES

Own or rent? Own the loop only where reliability is the product. Otherwise expect L2 and L7 to absorb it.

MOVES THAT WORK

- Invest in runtime assurance and recovery (L6e) rather than in more elaborate planning.
- Make human-in-the-loop (L6b) a designed product surface, not a fallback error path.
- Track task completion rate over long horizons; it is the only orchestration metric buyers repeat.

QUESTIONS TO ASK YOURSELF

1. Is my orchestration a differentiator or a feature the model layer will ship next release?
2. What happens on step seven of a ten-step task when step three was subtly wrong?
3. Who is accountable when the loop fails, and does the product make that person's job easier?

> **Characteristic failure mode**
>
> Building a general-purpose agent framework in a market that pays for one reliable workflow.

L7 — Surface

Interface, presentation, experience. How the user meets the intelligence.

People see the ring on the finger, the surface, the sparkle. If all you own is the display case (L7), anyone can build another display case. Embedded and transactional surfaces are the moats.

THE ANALOGY

Wearing the Jewelry, The Moment of Experience. People see the ring on the finger, the surface, the sparkle, the emotional moment. In AI: chat, dashboards, copilots, and ambient agents are the surfaces. Beautiful, but the most exposed layer, unless you're embedded inside the workflow or own the moment of transaction.

THE FIVE SUBLAYERS

Sublayer	What it covers
L7a Conversational	Voice and chat interfaces, the talking layer
L7b Visual Interfaces & Media	Dashboards, generated images, video, rich media output
L7c Embedded & Embodied AI □	AI woven into existing tools (IDE copilots, email assistants, in-app agents) and embodied in physical hardware (robots, devices, vehicles)
L7d Transaction Surface □	Where the AI closes a deal, books an appointment, processes a payment
L7e Async & Ambient Surfaces	Background agents, notifications, proactive nudges, always-on monitoring

Who plays here today	Structural verdict
ChatGPT, Gemini, Copilot, ElevenLabs	Modality = commodity. Context = moat.

OPERATOR NOTES

Own or rent? Own placement and habit; do not expect to own modality.

MOVES THAT WORK

- Chase the moment of consumption, not the elegance of the interface.
- Embed where the work already happens (L7c) instead of asking for a new destination.
- If you own a transaction surface (L7d), instrument it; that is where the surplus lands.

QUESTIONS TO ASK YOURSELF

1. Am I a destination the user must remember, or a surface already in front of them?
2. What is my default placement worth, and what would it cost a competitor to buy it?
3. If the interface were commoditized tomorrow, what would remain?

> **Characteristic failure mode**
> Winning a design award in a category the platform ships for free six months later.

L8 — Memory

Retention, learning, compounding context. What the system remembers.

The jeweler keeps records: which designs sold, which metals each customer prefers. Over time, this memory makes every decision better. Memory is the only layer that gets stronger every day.

THE ANALOGY

The Record Book, Compounding Knowledge. The jeweler keeps records: which designs sold, which metals each customer prefers. Over time, this memory makes every decision better. In AI: session, entity, network, institutional, and world-model memory compound. The system that remembers wins long-term.

THE FIVE SUBLAYERS

Sublayer	What it covers
L8a Session & Short-Term Memory	Within-conversation context, scratch state, working memory
L8b User & Entity Profiles	Persistent preferences, history, relationship context per user or account
L8c Aggregated Network Learning □	Patterns learned across many users/customers, fleet intelligence
L8d Institutional Knowledge □	What the organization knows, docs, decisions, tribal knowledge encoded
L8e Learned World Models □	The system's accumulated causal understanding of how things work

Who plays here today	Structural verdict
Sierra, Notion (partial), Rewind AI	The ultimate moat. Memory that compounds wins.

OPERATOR NOTES

Own or rent? Own it. Memory is the only asset that gets more valuable while you sleep.

MOVES THAT WORK

- Distinguish memory of events (defensible) from claims about truth (which inherit a regulator).
- Make institutional memory (L8d) exportable in principle and painful in practice — trust, then gravity.
- Aggregate learning across customers (L8c) only with contractual clarity; this is the fastest way to lose an enterprise account.

QUESTIONS TO ASK YOURSELF

1. Does my product measurably improve for a customer in month twelve versus month one?
2. What would a departing customer lose that they could not reconstruct?
3. Am I storing history, or compounding it?

> **Characteristic failure mode**
>
> Persisting a chat transcript and calling it memory.

Part III — The four laws

Four structural claims. They are stated as laws because they are meant to be falsifiable: a single well-documented counter-example mechanism forces an amendment, and amendments are published rather than quietly absorbed.

Law I — Intelligence Commoditizes Downward

> *If your product depends only on generic model capability, the platform layer below you will eventually absorb it. Wrappers don't survive, wrappers become features.*

Predicts WHO gets absorbed.

MECHANISM

Anything that can be done inside the model layer eventually will be, because the model owner controls the marginal cost. The wrapper does not lose on feature parity; it loses on price floor. You cannot charge a subscription for something the layer beneath you includes at zero marginal cost.

WORKED EXAMPLE

Jasper ($1.5B → ~$300M) was a wrapper on GPT. Once ChatGPT shipped, the value flowed to L2.

THE ESCAPE

Own a layer the platform structurally cannot: proprietary data, a trust gate, distribution, workflow depth, or compounding memory.

THREE QUESTIONS

1. Does my core loop require anything the model owner does not already have?
2. If the platform shipped my feature next month, what would my customers still need me for?
3. Is my pricing defensible against zero?

Law II — Value Accrues at Bottlenecks

> *Durable value rarely sits in the model or the UI. It sits at the scarce layer, proprietary data, workflow control, verification, distribution, memory, compliance, or trust. Find the bottleneck. Own it.*

Predicts WHERE value is going.

MECHANISM

Scarcity, not cost, determines margin. The model layer is expensive and abundantly supplied; those are different properties. Value settles wherever supply cannot expand as fast as demand.

WORKED EXAMPLE

NVIDIA owns L0 silicon. Vanta owns L3 compliance. Bloomberg owns L1b data. Each is the bottleneck in their chain.

THE ESCAPE

Name the bottleneck you own in one sentence. If you cannot, acquire or build one before scaling spend.

THREE QUESTIONS

1. What do I require that competitors cannot replicate within twelve months?
2. What do I require that the model layer is not incentivised to supply?
3. What would my customer have to rebuild if they left me?

Law III — The Surface Captures Attention; the Chain Captures Power

> *A beautiful UI may get users. But durable companies own a deeper layer of the intelligence chain, data, execution, memory, gates. Surface without depth rarely compounds.*

Predicts WHO survives the platform era.

MECHANISM

Surface wins the first cohort; chain wins the tenth. Interfaces are copied in weeks, and the platform copies them for free. What determines the future is which layers the product owns rather than rents.

WORKED EXAMPLE

Gamma owns L7 surface. Replit owns agent + code-gen + hosting + auth + database (L4 + L5 + L6 + L8). Same prompt-to-output category. Different fate.

THE ESCAPE

For every roadmap item, name the layer it reinforces. Items that reinforce only the surface are marketing, not strategy.

THREE QUESTIONS

1. Which layer does this quarter's roadmap actually strengthen?
2. If a larger surface entered my category tomorrow, what do I have that they would have to build rather than ship?
3. Am I accumulating anything while my users sleep?

Law IV — Generation and Verification Must Be Separate

> *Wherever output carries fiduciary, regulatory, safety, or reputational weight, the generator and the verifier must be separate economic entities. L3 above L2/L5 is structurally permanent, the model can't audit itself, the codegen can't certify itself, the drafter can't approve itself.*

Predicts WHERE L3 is non-absorbable.

MECHANISM

Self-attestation carries no information to a third party, so verification must be performed by a separate economic entity. This is institutional, not technical, and institutional constraints harden rather than soften.

WORKED EXAMPLE

Vanta (L3) over AWS/OpenAI. Snyk (L3) over Copilot. Big-4 audit over SAP. Ironclad over Harvey. The verifier survives every model generation.

THE ESCAPE

If you generate in a regulated domain, do not fight the verifier above you. Become the preferred generator routed through it.

THREE QUESTIONS

1. Whose signature is on the line when my output is wrong?
2. Would my buyer's auditor accept my own logs as evidence?
3. Is there a regulator, insurer, or board that already requires a second party?

The four laws on one card

	Says	Predicts	Example
I	Intelligence commoditizes downward	WHO gets absorbed.	Jasper ($1.5B → ~$300M) was a wrapper on GPT. Once ChatGPT shipped, the value flowed to L2.
II	Value accrues at bottlenecks	WHERE value is going.	NVIDIA owns L0 silicon. Vanta owns L3 compliance. Bloomberg owns L1b data. Each is the bottleneck in their chain.
III	Surface captures attention, chain captures power	WHO survives the platform era.	Gamma owns L7 surface. Replit owns agent + code-gen + hosting + auth + database (L4 + L5 + L6 + L8). Same prompt-to-output category. Different fate.
IV	Generation and verification must be separate	WHERE L3 is non-absorbable.	Vanta (L3) over AWS/OpenAI. Snyk (L3) over Copilot. Big-4 audit over SAP. Ironclad over Harvey. The verifier survives every model generation.

Part IV — The three currents

The layers are supply-side. The currents are market forces flowing across every layer, and they decide whether a defensible position becomes a business. A company can be perfectly positioned on the map and still starve if no current is running through its layer.

Current I — Demand Gravity

Where the budget actually sits, and what it pulls toward.

As L2 prices collapse, discretionary spend migrates from buying generation to buying outcomes (L5 + L8), verification (L3), and access to proprietary data (L1). A defensible layer with no buyer is worth zero.

> **How to use it**
> Name the buyer, the budget line, and the line item they stop paying for once L2 is free.

Current II — Attention Economics

What becomes scarce when generation becomes infinite.

Default placement, OS integration, habit loops and on-ramp ownership decide which intelligence is actually used. The large surfaces operate as landlords charging rent in attention. Law III names the asymmetry; this Current prices it.

> **How to use it**
> Assume infinite supply. Ask who owns the on-ramp and what default placement costs.

Current III — Capital Flows

How funding rounds bend the layers they fund.

Capital is reflexive. Tens of billions into L2 produced a generation glut; near-zero into L-1 produced the physical bottleneck now constraining everything above it. Capital overheats the fashionable layer and starves the unglamorous one.

> **How to use it**
> Read the funding map as a distortion field, not as a value signal.

Reading the currents together

Signal	Read
One current pointing at your layer	A tailwind. Necessary, not sufficient.
Two currents	A real market forming. This is the moment to over-invest.
Three currents	A category. Expect capital, incumbents, and platform attention within four quarters.
No current	A well-defended position nobody is paying for. The most expensive failure mode there is.

Note what is deliberately absent. Geopolitics is not a current: it acts at physical resources and at gatekeeping, and promoting it to a horizontal force double-counts the same effect twice. Keeping the list at three is a discipline, not an oversight.

Part V — Dynamics

The laws say what is structurally true. The dynamics describe how the market actually moves: repeatable patterns observed across hundreds of companies, and the six positions companies settle into. Patterns can earn promotion to laws over time; none of these has yet.

Pattern 1 — The Two-Vendor Rule

Layers touched: L3, L5

Enterprises will pay for two vendors when one vendor's mistake is unrecoverable. Codegen + code-security. Draft + review. Model + eval. Trade + clearing. The buyer pays the duplication tax to avoid the single-point-of-failure tax.

- Cursor for codegen + Snyk/Semgrep for security review, no CISO accepts the same vendor doing both.
- Harvey drafts contracts; Ironclad/Kira reviews them. The drafter is structurally not allowed to be the approver.

Pattern 2 — Regulatory Half-Life

Layers touched: L3

The more regulated the industry, the longer L3 outlives L2 churn. A compliance gate written into law is a moat measured in decades, not quarters. Models cycle every 6 months; SOC 2, HIPAA, EU AI Act, FDA 510(k) cycle every 5–10 years.

- Vanta and Drata are 4 model generations old and untouched. The frontier model labs are not certifying themselves.
- Epic's L3+L4 position in healthcare predates the entire AI wave and will outlive GPT-7.

Pattern 3 — The Bundling Asymmetry

Layers touched: L2, L3

Foundation model labs will expand from L2 into L5/L6/L7, adjacent value, because the buyer accepts the same vendor doing both. They will not expand across the trust boundary into L3 above themselves. OpenAI will ship agents. OpenAI will not issue its own SOC 2 audit.

- OpenAI shipped GPTs, the Apps SDK, Operator, and Codex, all L5/L6/L7 expansion. None of it is self-certification.
- AWS ships hundreds of services but pays Vanta/Drata for compliance evidence. The platform respects the boundary.

Pattern 4 — Memory Is Not Truth

Layers touched: L8, L3

L8 memory of what happened, what the user said, did, preferred, is a clean moat. L8 claims about what is true, diagnoses, legal positions, financial valuations, require an L3 verifier above them. The moment memory makes a truth claim, it inherits a regulator.

- Notion AI remembers your docs (L8b, defensible). It does not diagnose your patients.
- An AI medical scribe (L8) is valuable; the same scribe issuing a diagnosis triggers FDA (L3) and an MD signature requirement.

Pattern 5 — Distribution Eats Generation

Layers touched: L2, L7

Once L2 commoditizes (and it always does), the surplus flows to whichever layer owns the user's moment of consumption, L7c (embedded copilot) or L7d (transaction surface). The model is generic; the context of use is not.

- Cursor captures the codegen surplus, not the model underneath it. The model is interchangeable; the IDE moment is not.
- Perplexity captures the answer surplus by owning the question moment. The model could be any of four, the surface is the moat.

Pattern 6 — The Gatekeeper Tax is Always Arbitraged

Layers touched: L1, L3, L7

Wherever a gatekeeper extracts rent between the marginal cost of supply and the perceived value of demand, an arbitrageur, API shim, cloud automation, open-source replacement, lateral integration, or regulatory appeal, will step into the gap. The gatekeeper's pricing power is bounded by the cost of the workaround. The arbitrageur lives at L7 and quietly reaches down into L5 to widen the margin further.

- Dripify (L7) arbitrages LinkedIn's (L1+L3) connection-request bottleneck: cloud automation + proxies cost pennies, sales teams pay $39–$99/seat/month. Newer entrants now hook L5 open-source LLMs to auto-reply, compressing the last human cost.
- Plaid (L4b) arbitraged the bank gatekeepers' API absence for a decade; the moment banks shipped their own APIs, Plaid's margin compressed and it had to migrate up into identity and data.

The six archetypes

Over a long enough horizon, most software companies settle into one of six positions. The archetype is not destiny, but moving between them requires acquiring a layer, not shipping a feature.

Archetype	Status	Structure	Illustrative
Data Refineries	Safe	L1b proprietary data compounds faster than competitors can accumulate it.	Apollo, Bloomberg
Infrastructure Rails	Safe	L4b / L4e — essential pipes and agent identity, boring and load-bearing.	Supabase, Twilio
Workflow Fortresses	Contested	L5 + L6b — the system of record plus human-in-the-loop control.	Salesforce, HubSpot
Domain Specialists	Safe	L5a/b/d plus L8c — encoded expertise that improves with use.	Harvey, Sierra
Thin-Layer Surfaces	Exposed	L7a / L7b with no deeper layer owned. Absorbed on the platform's schedule.	Jasper (2023), most 2024 wrappers
Full-Stack Juggernauts	Dominant	L2a plus L7c/d plus L8c — generation, surface and memory in one entity.	ChatGPT, Copilot, Gemini

The two useful questions are: which archetype am I today, and which one am I becoming? A workflow fortress drifting toward thin surface is the most common and least noticed trajectory in enterprise software, because the drift shows up as good quarterly news — more users, more sessions — while the underlying layer ownership erodes.

Part VI — Instruments

Instrument 1: the Defensible Triangle

The triangle is L1b proprietary data, plus L5a/b/d deep execution and playbooks, plus L8c/d/e compounding memory. It is the most common fortress pattern among application-layer companies, and it works because each vertex makes the others harder to copy: proprietary data makes execution better, execution generates outcome data, and memory turns both into something that improves with age.

Vertex	What it is	How you know you have it
L1b Proprietary data	Data behind a wall nobody else can reach.	A competitor with your exact UI would still need years to accumulate it.
L5a/b/d Deep execution	Domain judgement, decision frameworks, encoded SOPs.	A generalist with a frontier model could not reproduce 80% of it in a weekend.
L8c/d/e Compounding memory	Network learning, institutional knowledge, learned models.	The product is measurably better for a customer at month twelve than at month one.

The triangle is a pattern, not a requirement. A pure gatekeeper wins on L3 alone; a silicon vendor wins on L0 alone; a data platform wins on L4 alone. Owning one layer deeply is sufficient. What is never sufficient is owning a thin sliver of a contested layer.

Instrument 2: the Intelligence Cube

The cube places a company in three dimensions at once: corporate function on one axis, industry vertical on another, and layers on the third. Volume is the read. A company occupying a large contiguous volume — several layers deep, across a coherent function and vertical — is structurally durable. A thin sliver, one layer across one function, is not, no matter how good the product is.

Axis	Values
Function	Engineering, Design, Product, Project management, Operations, Marketing, Sales, Customer care, Strategy
Vertical	Financial services, Education, Legal, Health, Travel, Commerce, Media, Government, Software, Horizontal
Layer	L-1 through L8, counting only layers the company actually owns rather than rents

Two practical uses. First, whitespace: plot the occupied cells in a vertical and the empty ones are candidate positions, though an empty cell is more often a bad market than an oversight. Second, expansion sequencing: adding depth in one cell almost always beats

adding a second thin cell, because depth compounds and breadth divides attention.

Instrument 3: the defensibility audit

Eight questions. Score each from 0 to 12 where 0 is a flat no and 12 is an unambiguous yes with evidence you could show a sceptical board member. Total out of 96, normalised to 100. Be harsh; the audit is only useful if it can produce a bad score.

Area	Question	Layer
Model dependency	Could a better GPT/Claude/Gemini release replace your core value?	L2
Data ownership	Do you create or own proprietary context that competitors can't access?	L1b
Workflow depth	Are you embedded in a daily or high-stakes workflow users can't easily exit?	L5 / L6
Trust gate	Do users rely on you for verification, compliance, quality, or approval?	L3
Distribution	Do you own a channel, community, brand, or enterprise relationship?	L4 / L7c
Memory	Does the product become smarter or more useful with usage history?	L8
Switching cost	Would leaving you destroy useful state, process, or institutional knowledge?	L8d
Platform exposure	Could a major platform (OpenAI, Google, Microsoft, Salesforce) bundle this for free?	L2 / L7

Score	Band	Read
0–20	Thin Wrapper	Generic model + thin UI. The platform will absorb you.
21–40	Useful Tool, Weak Moat	Real utility, but no structural protection. Time-bound.
41–60	Workflow Product	Embedded in a workflow. Survivable, but watch the platforms.
61–80	Defensible AI System	Owns multiple layers. The chain is yours, not rented.
81–100	Intelligence Gate	Platform candidate. You are the bottleneck others must cross.

How to use a bad score

A low score is not a verdict on the product; it is a statement about which layers you rent. The remedy is always the same shape: pick one layer you can credibly own within four quarters, and make the next two roadmap cycles about acquiring it rather than about surface improvements. Surface work is easier to ship, always demos better, and does not move this score.

Instrument 4: the agent decoder

The word “agent” is not a layer. It is packaging. Decoding it before analysis is the single highest-return habit in this framework, because the word hides exactly the information you need.

Component	Layer	Required?
The skill — doing the actual work	L5 Execution	Always
Multi-step planning, routing, tool use	L6 Orchestration	Required for anything called agentic
The interface it is met through	L7 Surface	Usually
Cross-session recall	L8 Memory	If it remembers
Connectors, permissions, protocols	L4 Access	Ridden, never owned by the agent

The common analytical error is tagging an agent product as an access-layer play because it uses connectors. It rides those pipes; it does not own them. Name the execution and orchestration first, then say which of access, surface, and memory it bundles. If a product claims to be an agent and you cannot name its L5, it is a surface with a loop.

Part VII — Worked readings

Six readings, each in the same shape: what the company appears to be, what it actually owns, what it rents, and the structural conclusion. Occupancy changes; the method does not. Run the same shape on your own company before you argue with the conclusions.

1. The absorbed wrapper

Appears to be: a category-defining AI writing product with strong brand and rapid early growth. **Actually owns:** a surface and a set of templates. **Rents:** the model, the data, the distribution. **Conclusion:** when the model layer shipped an equivalent loop inside a surface users already had open, the price floor went to zero and the valuation followed. Nothing about the product got worse. The layer beneath it absorbed what it was charging for. This is Law I in its purest observable form, and it is the base case against which every application-layer position should be argued.

2. The bottleneck owner

Appears to be: a component vendor in a cyclical hardware market. **Actually owns:** the scarce input to every other layer, plus the software ecosystem that makes switching expensive. **Rents:** fabrication, which is the one genuine exposure. **Conclusion:** pricing power follows scarcity, not visibility. The layer that looks least like a business in a boom is frequently the one collecting the boom's margin. Read the physical-resources layer beneath it for the constraint that actually binds.

3. The gatekeeper above the giant

Appears to be: a compliance automation tool, technically unremarkable next to the infrastructure it sits above. **Actually owns:** a trust gate that the infrastructure provider cannot occupy for itself. **Rents:** everything technical. **Conclusion:** the most capable firm in the chain pays a less capable firm for attestation, and will continue to, because the constraint is institutional. This is the clearest live illustration of Law IV, and the reason verification positions survive model generations that erase adjacent products.

4. Same category, different chain

Appears to be: two prompt-to-output products with comparable interfaces and comparable early traction. **Actually owns:** one holds only the surface; the other holds access, execution, orchestration, and memory. **Conclusion:** feature parity is not structural parity. The interfaces converge; the futures diverge. When you benchmark competitors, benchmark layers owned, not features shipped.

5. The data refinery

Appears to be: a mature information business in a category everyone assumes AI will disrupt. **Actually owns:** decades of structured proprietary data plus the workflow in which it is consumed. **Rents:** the model, cheerfully and interchangeably. **Conclusion:** where the corpus is the moat, model progress is a tailwind rather than a threat. Every capability improvement makes the data more valuable, because the data is the part that cannot be synthesised.

6. The arbitrageur

Appears to be: a small tool with unremarkable technology and remarkable margins. **Actually owns:** a position in the gap between a gatekeeper's marginal cost and the value the gatekeeper's constraint creates. **Conclusion:** every gatekeeper margin attracts an arbitrageur, and the arbitrageur's ceiling is the cost of the workaround. These businesses are real, often very profitable, and structurally temporary. Price them accordingly, and do not confuse the margin for a moat.

Part VIII — Applications

1. Building a roadmap that moves your position

Most roadmaps are lists of features grouped by customer request. A layer-aware roadmap groups by the layer each item reinforces, which makes an uncomfortable pattern immediately visible: the majority of shipped work usually reinforces the surface, because surface work is faster, demos better, and is what customers can articulate.

Roadmap item	Layer reinforced	Compounds?
A new template library	Thin L5	No
A visual refresh	L7	No
An integration that captures usage data competitors cannot see	L1c + L8	Yes
A compliance certification your buyers require	L3	Yes
Write-back into the customer's system of record	L4	Yes
Encoding a customer's approval rules as a configurable playbook	L5d + L8d	Yes

1. Tag every roadmap item with the layer it reinforces. Disallow “multiple”.
2. Compute the share of engineering weeks going to compounding layers. Most teams find it below twenty percent.
3. Set a floor — a third is a reasonable starting point — and protect it through the quarter, because it will be the first thing traded away.
4. Re-run the audit in Part VI every two quarters. If the score has not moved, the roadmap was surface work with good release notes.

2. Diligence: reading a company in thirty minutes

A structured pass that produces a defensible view quickly. It will not tell you whether the company wins. It tells you what has to be true for it to win, which is the more useful output of a first meeting.

1. **Decode the pitch.** Strip the words agent, copilot, and platform. Write what the product does in one sentence a customer would recognise.
2. **Tag the layers owned.** Not touched, not integrated with — owned. Two or three is normal; more than four claimed usually means fewer than two real.
3. **Tag the layers rented.** For each, ask what happens if the supplier raises price, changes terms, or ships the same feature.

4. **Locate the bottleneck.** One sentence naming what competitors must pay to cross. If the founder cannot supply it, that is the finding.
5. **Check the currents.** Who is the buyer, what budget line, and what do they stop paying for once generation is free?
6. **Apply Law I.** Name the specific platform release that would compress this company, and estimate how many quarters away it is.
7. **Apply Law IV.** If the output carries weight, is this the generator or the verifier? Generators in regulated domains that pitch themselves as verifiers are mispricing a structural constraint.
8. **Place the archetype.** Which of the six is it today, and which is it becoming?

> **The single most useful diligence question**
> "What would a well-funded competitor with a frontier model and your exact interface still not have twelve months from now?" The quality of the answer is close to a sufficient statistic for the quality of the position.

3. A reference architecture for agent systems

The framework was built to analyse markets, but it works as an architecture checklist because the layers are also the components. Reading an agent design as a chain surfaces the omissions that show up in production three months later.

Layer	Design decision	Common omission
L1 Data	What context does the agent see that competitors' cannot?	Grounding entirely in public or customer-supplied documents
L2 Models	Routing policy: which task goes to which model at which price?	One frontier model for every call, including trivial ones
L3 Gates	What must be checked before an action is allowed to commit?	Evaluation treated as a pre-launch activity rather than a runtime gate
L4 Access	Which systems can it read, which can it write, under whose credentials?	Agent acting under a shared service account with no per-action provenance
L5 Execution	What judgement is encoded, and by whom was it reviewed?	Prompt engineering in place of an encoded decision framework
L6 Orchestration	What happens at step seven when step three was subtly wrong?	No recovery path, no checkpointing, no human escalation surface
L7 Surface	Where does the human meet it, and is that where the work already is?	A new destination the user must remember to visit
L8 Memory	What persists, for whom, and what improves because of it?	Transcript storage described as memory

Two design rules follow directly from the laws. First, keep generation and verification in separate components with separate owners wherever an action carries weight — the same architectural discipline that Law IV describes at market scale applies inside a single system, and for the same reason. Second, design so that ordinary operation produces outcome data: an agent that records what happened after it acted is accumulating an asset; one that records only what it produced is accumulating logs.

4. Building a market map

A market map places real companies on the ten layers within one vertical. It is the most effective way to learn the framework, and it produces something genuinely useful to other people, which is rare in strategy work.

1. Pick a vertical narrow enough that you can name thirty companies without searching.
2. Build the grid: ten layers by five sublayers. Fifty cells.
3. Place each company in the cells it *owns*. Most companies occupy two or three; if you are placing one in ten, you are recording marketing rather than structure.
4. Mark the empty cells. Ask, for each, whether it is whitespace or a bad market. Most are bad markets, and saying so is the valuable part.
5. Write the thesis paragraph last: where value is concentrating in this vertical, and which current is moving it.

Two disciplines make maps trustworthy. Name a real company in every claim — abstract archetypes are lazy and unfalsifiable. And date the map, because occupancy changes weekly while the structure does not; an undated map will be quoted long after it is wrong.

Part IX — Glossary

Term	Definition
Bottleneck	A layer where supply cannot expand as fast as demand. Where margin settles.
Compounding memory	L8c–e. State that makes the product more valuable to a specific customer over time.
Current	A market force flowing horizontally across all layers. There are exactly three.
Defensible Triangle	L1b + L5a/b/d + L8c/d/e. The most common application-layer fortress pattern.
Generation	Production of a candidate output. Distinct from verification.
Intelligence Cube	Function × vertical × layer. Volume indicates structural durability.
Layer	One of ten positions in the production chain, L-1 through L8.
Outcome data	L1d. What happened after the model acted. The scarcest and most defensible data class.
Register	Substrate, Workflow, or Surface. A grouping of layers by half-life.
Sublayer	One of five subdivisions within a layer. Fifty in total. Most moats live at this level.
Surface	L7. What the user touches. Modality commoditizes; placement and habit do not.
Thin sliver	A position occupying one contested layer and no others. The weakest position on the map.
Two-vendor rule	Buyers pay for two vendors when one vendor's error is unrecoverable.
Verification	An assertion to a third party that an output meets a standard. Cannot be credibly self-issued.
Wrapper	A product whose value derives only from generic model capability. Absorbed on the platform's schedule.

Companion documents

- **Academic theory paper (v4.1)** — the single theoretical claim, its relationship to prior literature, and six falsifiable predictions.
- **Working paper (19 pages)** — literature review, theory, vignettes, alternative explanations, and research agenda.
- **This practitioner guide** — the full taxonomy, instruments, and applications.

How to cite

Arivukkarasu, A. (2026). *Supply Chain of Intelligence: where competitive advantage accumulates in AI markets* (Version 2.0). https://supplychainofai.com/papers

BibTeX: @misc{arivukkarasu2026scoi, author = {Arivukkarasu, Anand}, title = {Supply Chain of Intelligence}, year = {2026}, note = {Version 2.0}, url = {https://supplychainofai.com/papers}}