SINGULARICS

White Paper

The Domain Knowledge Network

A new layer of the organization. A governing structure that owns correctness, not a help desk.

Alexander S. Petty
SINGULARICS | singularics.ai

First published August 2026.

singularics.ai

Contents

Abstract

Every prior operating framework assumed that domain knowledge was a given. Teams were expected to either hold it already or find it when they needed it. That assumption held as long as execution was the constraint, because the pace of delivery gave domain experts time to be found. When AI makes execution cheap and fast, domain context and timely judgment become the scarce resources, and scarce resources need structure. The Domain Knowledge Network is that structure. It owns correctness across a portfolio through a governed guild of floating experts who inject knowledge before work begins, encode their criteria into the verification harness so that judgment runs on every push, and return what they learn from the field to the network itself. This paper defines the network in full: its membership, its operating rule, how injection works as a push model rather than a pull model, why encoding is the step that makes a small guild viable, and why this layer has no precedent in prior frameworks.

The Shape of It

In every portfolio we have worked with, the same pattern repeats. The first team to touch a policy area spends two days finding the right domain expert and three hours getting the constraints articulated. Six weeks later, a second team touches the same policy area and spends the same two days and three hours, because nothing from the first conversation persisted in a form anyone else could use. Multiply that across a portfolio of twenty active efforts and the waste is measured in months, not days.

A governing structure, not a help desk. It owns correctness, and correctness has two faces. Right against the rules, and right about what customers value. The Decision Owner owns the outcome.

That distinction carries weight. In most organizations today, domain expertise is consumed reactively. A team hits a question it cannot answer, searches for the right person, waits for availability, and receives an answer that lives in the memory of whoever was in the room. The knowledge is consumed but not retained. The next team with the same question starts from zero.

The Domain Knowledge Network replaces that pattern with a governed layer. Experts are named, their commitment is funded, their contributions are encoded into lasting artifacts, and their availability is planned rather than begged for.

> *It is a new layer of the organization, and it exists because when execution stops being the constraint, domain context becomes the scarce resource. Scarce resources need structure.*

Machine speed raises the stakes from efficiency to survival. A fleet propagates whatever enters it with perfect fidelity, including a wrong assumption, so the only place to catch an error cheaply is before the build, which is exactly where the network stands.

The network sits between the portfolio governance bodies and the teams. The Intent Council sets direction and standards. The Flow Council runs the work day to day. The Domain Knowledge Network ensures that the work is right against policy, domain rules, and regulation, and honest about what customers value. That is its accountability, and it is distinct from both the team's accountability for delivery and the Decision Owner's accountability for the outcome.

Three things are owned in this model, and not by the same party. The Decision Owner owns the outcome. The team owns the effort. The Domain Knowledge Network owns correctness, right against the rules and right about what customers value. Keeping those three separate prevents the most common confusion.

Membership

The network draws from every function that holds institutional knowledge: policy, compliance, risk, data, operations, business analysts, and the customer-facing people in research, service, support, and sales who carry the evidence of what customers value. Members remain in their home functions. Network membership is an additional, named, funded commitment.

20–30% of each member's week is committed to the network. This is funded time, not volunteered time. Without funding the commitment erodes within weeks.

The seats on the guild are knowledge streams rather than job titles, and people map onto them in combinations. One person often carries two or three streams, a compliance lead who also holds policy, an analyst who holds domain and data. A stream may have several holders. The floor is that every stream has at least one

named holder, and no stream depends on exactly one person for long. A small guild of three or four people can cover every stream between them.

Membership is not honorary and it is not rotational. Named individuals are selected because they carry specific domain knowledge the portfolio needs. They are known to the Flow Council, scheduled into the weekly capacity plan alongside every other resource, and evaluated in part on the quality of their contributions to teams they are not members of.

The guild is deliberately cross-functional. A single-discipline network (all policy experts, for example) cannot govern correctness across a portfolio that touches pricing, operations, compliance, data, and policy in the same week. The power of the network is its breadth, and breadth requires representation from every function that can make a team wrong.

The size of the network scales with the portfolio. A portfolio running three to five simultaneous efforts needs eight to twelve network members. Larger portfolios need more, but the ratio of network members to active efforts should stay roughly two to one, because each effort will need access to more than one specialty over its lifetime.

The strongest objection to the network is cost. “We cannot afford to pull twelve domain experts away from their day jobs for 20 to 30% of their week.” The math runs the other way. Without the network, each of those experts is already being pulled away, reactively, for troubleshooting after teams stall. The reactive model consumes the same hours but delivers no encoding, no compounding, and no reduction in future demand. The network does not add a cost. It converts an invisible, unmanaged cost into a visible, declining one.

The One Rule

> *Every expert floats and none are dedicated. They go where knowledge is needed today.*

This is the operating rule that separates the network from a staffing model. In a staffing model, a domain expert is assigned to a team for the life of the effort. That creates depth for one team and starvation for every other team that needs the same specialty. In a portfolio with more efforts than experts, which is the normal condition, dedication is arithmetic that does not close.

Floating solves the arithmetic. A policy expert who spends Tuesday morning with one team and Wednesday afternoon with another covers two efforts with 20% of their week. The contribution is short, targeted, and high-value because the expert arrives with a specific question to answer rather than sitting through a build day waiting to be useful.

The Decision Owner is the *only* exception. The role is filled from this same business guild and sits with one team for most of the week, because someone has to be present to accept every single day. The rest of the owner's week stays in the business on purpose. The knowledge that made them worth naming is perishable, and it is refreshed only by doing the business. That standing daily presence is what distinguishes the Decision Owner from the rest of the network. They share the same domain depth, but they carry a different obligation: ownership of the outcome rather than governance of correctness.

State this rule with force, because the pressure to dedicate experts is constant. Every team lead prefers a dedicated expert because it removes waiting. But the portfolio cannot afford it. A guild of twelve experts serving six efforts can float. Dedicating two per effort requires twelve full-time commitments from people whose home functions need them 70 to 80% of the time. The arithmetic is obvious once stated, and it is ignored the moment it is convenient. The Flow Council holds this rule at weekly calibration, and it holds it against every team that asks for an exception.

The Decision Owner Is the First Line

Because the Decision Owner is drawn from this guild and stays with the team, the team already holds domain knowledge for its day-to-day decisions. That is the point of embedding them. The team proceeds without waiting on anyone for knowledge its own owner carries.

The network exists for reach beyond what the owner carries. An effort owned by a pricing expert that runs into a disclosure rule needs a specialist the owner is not. An operations outcome that hits a policy constraint needs someone from compliance who understands the regulation in question. These are the moments the network serves.

> *If a team is calling the network for questions its own Decision Owner could answer, the wrong person was named as owner. That is a naming failure, not a network failure, and it is fixed at formation.*

This test is diagnostic. Track how often each team calls the network and what category of question triggers the call. If the pattern shows that the team's domain needs fall consistently outside the owner's expertise, the effort was assigned to the wrong owner at formation. The correction is to replace the owner, not to increase the network's allocation.

A well-named Decision Owner handles 80 to 90 percent of the domain questions a team encounters. The network handles the remaining 10 to 20 percent, which are the cross-cutting, specialist, or edge-case questions that no single person is expected to carry.

The decision rule at formation: if the effort's first two weeks of specs will require domain knowledge the proposed Decision Owner does not carry, either name a different owner or pre-schedule the relevant network members for those weeks. Do not wait for the team to discover the gap during the build. That discovery costs a full day per occurrence.

How Injection Works

The network operates on a push model, not a pull model.

Each morning, the network reads what the day's specs require and connects the right expert to the right team before anyone has to ask. The signal for this comes from the previous evening's daily planning, which names the domain expert tomorrow's spec will need.

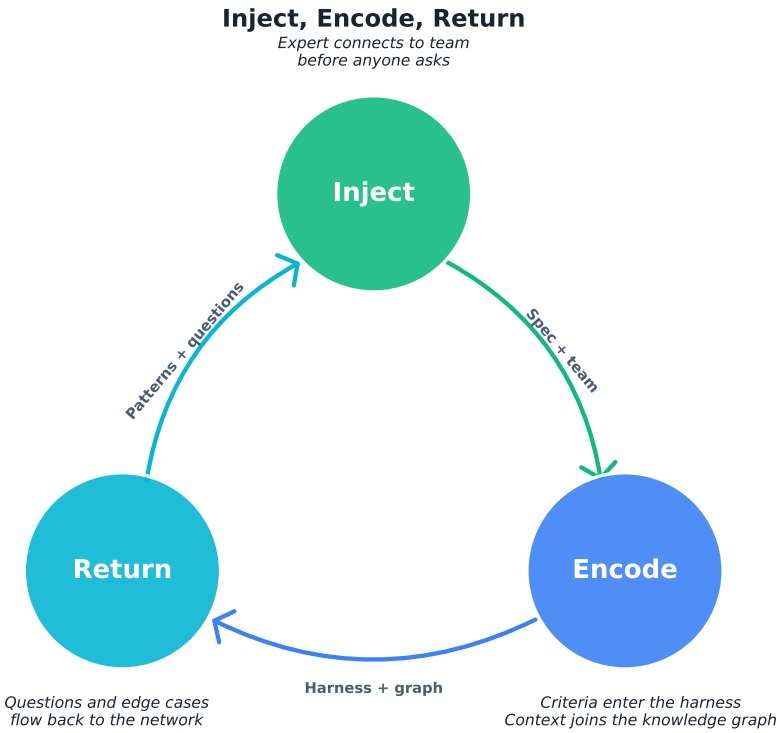

Figure 1: The three-phase cycle of the Domain Knowledge Network. Knowledge is injected before the build starts, encoded into the harness and knowledge graph so it persists, and questions from teams return to the network where they update the shared context. The cycle compounds: each rotation reduces the need for the next injection.

A typical contribution is a 15 to 30 minute working session. The specialist hands over the policy constraints and validation criteria the work must satisfy. This is not a review cycle. It is not a gate. It is a transfer of knowledge that happens before the build begins, so the team builds right the first time rather than building, reviewing, correcting, and building again.

> *Governance arrives as knowledge on the way in, rather than as a review queue on the way out.*

The push model inverts the traditional pattern. In a pull model, teams discover they need domain knowledge when they hit a wall. They search for the right person, wait for availability, and lose hours or days. In a push model, the network anticipates the need from the spec and connects the expert before the wall is hit.

The signal mechanism is lightweight and daily. Daily planning already names what the next increment will touch and which domain expert tomorrow's spec will need. A network coordinator reads those signals each evening and dispatches from the guild. By morning, the expert knows which team to join, what question to answer, and what spec to read before the session.

The contribution itself is a 15 to 30 minute working session. Not a review cycle. Not a co-authoring session. The expert hands over the constraints, the validation criteria, the edge cases, and any policy context the spec must satisfy. The Decision Owner and the Fleet Lead absorb it, and the build day proceeds. The expert returns to their home function until the network dispatches them again.

No ticket system. No request queue. No approval workflow. The coordination cost is one read of the daily planning outputs by the coordinator each evening, and one short notification to the matched expert. If the coordination is heavier than that, the mechanism has been over-engineered.

15–30 min per injection session. If an expert is spending more than 30 minutes with a team, either the spec touched more domain surface than anticipated (split the increment) or the expert is co-authoring rather than validating (return to the authorship split: agent drafts, owner signs, expert validates).

Encode

Injection alone does not scale. A guild of part-time experts cannot attend every team's every cycle, and if every contribution lives only in the memory of whoever was in the room, the network is consulting rather than governing. The same question returns next month.

The step that makes a floating guild viable is not injection but what happens immediately after it. The expert's criteria are encoded into the harness, and their context joins the knowledge graph. From that moment, their judgment runs on every push, for every team, without the expert in the room.

This is what makes the layer workable. Each contribution leaves executable checks and reusable context behind, so demand for specialist time falls as the portfolio learns. In practice, a well-run network reduces expert hours per outcome by 40 to 60% over the first six months, because the harness carries forward every constraint that was previously encoded.

Rule: Encode, not merely explain. If nothing outlives the conversation, the network is consulting rather than governing.

Encoding stores enforcement, and it does not replace the expert. A check enforces what was foreseen. The guild notices what was not, and noticing is the part that stays human.

Consider a policy expert who injects a set of constraints into a team's spec. Without encoding, those constraints live in the spec document and in the memories of the people who heard them. With encoding, those constraints become checks in the harness that fire on every push. The next team that touches the same policy area inherits those checks automatically. The expert never has to explain the same constraint twice.

The knowledge graph serves a parallel function for context that is not directly checkable. Institutional knowledge, edge cases, policy interpretations, and historical decisions are captured in a structured form that the spec agent can surface when relevant. The expert's contribution becomes part of the organizational memory rather than part of one team's story.

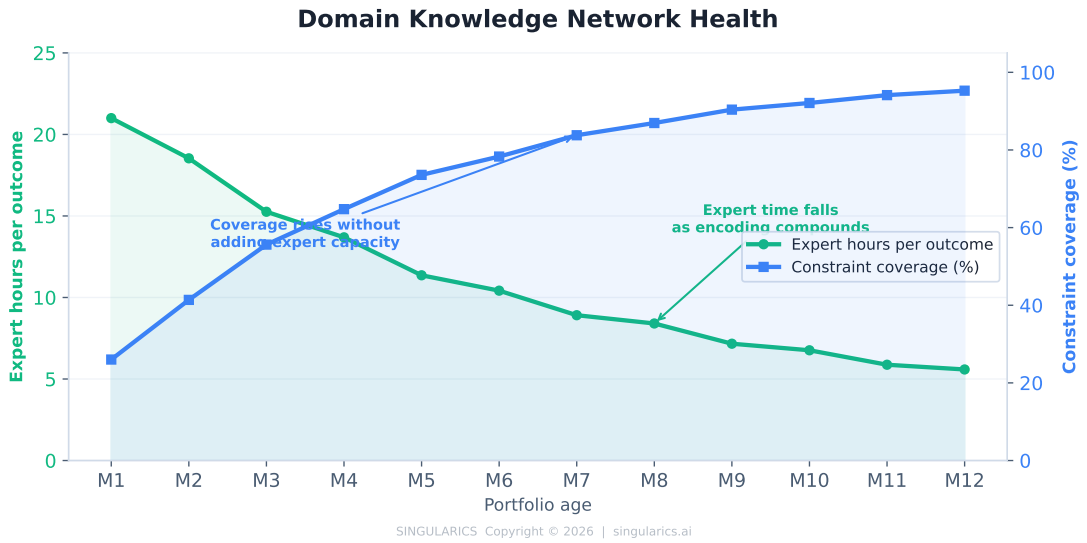

Figure 2: The health test of the Domain Knowledge Network. Expert hours per outcome trend down as encoding compounds. Constraint coverage trends up as each contribution leaves executable checks behind. A network where both lines are flat is explaining but not encoding.

The health test

The health test of the network is visible in two metrics, and both must move together.

↓ Expert hours per outcome trending down. Each contribution that is properly encoded reduces the need for the next one. If expert hours per outcome are flat over a quarter, the network is explaining but not encoding, and the same questions are returning.

↑ Coverage of constraints trending up. Each harness check that encodes an expert's criteria is a constraint that runs on every push without the expert in the room. If coverage is flat, the network is not reaching new territory.

Both lines moving in the right direction is the signal that the network is compounding. If expert hours are falling but coverage is also falling, the network is disengaging rather than encoding. If coverage is rising but expert hours are also

rising, the encoding is not reducing demand, which usually means the checks are too narrow or the knowledge graph is not surfacing context to spec agents effectively.

The rota itself is set at weekly Flow Council calibration, so expert availability is planned alongside capacity rather than begged for in the moment. The health test metrics are reviewed at the same calibration.

How the Network Learns

Knowledge does not flow in only one direction. Questions, edge cases, and new patterns flow back from teams into the network, where they update the playbooks the next team will receive.

This return flow is what makes the network compound rather than merely serve. Every team that encounters a new edge case, discovers a policy ambiguity, or finds a constraint the network had not anticipated is generating knowledge that belongs to the network rather than to the team.

Without a mechanism to capture it, that knowledge disperses. The team remembers. The next team does not.

> *Budget a monthly two-hour network session for absorbing what came back and revising the shared context. Without this, the network becomes a rota of interruptions.*

The monthly session has a specific structure. Network members review the questions and edge cases that arrived since the last session, categorize them by domain, and decide three things for each: whether the existing playbook already covers it and the team simply did not find it, whether the playbook needs to be updated, or whether the pattern represents something new that requires a policy clarification from the Intent Council.

The first category is a signal problem. The knowledge existed but the team could not access it, which means the knowledge graph needs better indexing or the spec agent needs better retrieval.

The second category is the network doing its primary work. The playbook grows, the harness gains new checks, and the next team benefits.

The third category is rare and important. It surfaces questions that the network itself cannot answer because they require a decision from the officers. These are

escalated cleanly, with evidence from the field, rather than arriving as abstract policy debates.

Over time, the monthly session becomes the primary mechanism by which the organization's institutional knowledge is maintained, updated, and made accessible. It replaces the informal hallway conversations and tribal knowledge that every organization relies on and none can audit.

The Customer Stream

Most of the guild guards correctness against the rules, and rules move slowly. A policy constraint encoded last year is probably still a constraint. The customer seat is different in kind. What customers value decays in months rather than years, and it is the one stream where an organization is most tempted to substitute its own consensus for evidence. Every enterprise believes it knows its customer. What it usually knows is what its customer wanted the last time anyone honestly checked.

The network treats customer knowledge with the same three movements, plus one added discipline. Evidence of what customers value enters the outcome brief when the brief is authored, cited and dated, the way a policy constraint enters a spec. It is encoded into the knowledge graph with its date attached, so the next brief can tell current knowledge from last year's customer. And what live work surfaces, support themes, usage patterns, acceptance evidence, returns to the network like any other edge case.

> *Evidence, not assumption. If nobody can point to where a customer said it, showed it, or did it, the brief records an assumption to be tested rather than a fact to build on.*

This mattered less when a wrong guess about the customer had months of meetings to get caught in. Now the fleet executes the guess with perfect fidelity, the same day, and the harness proves that it did. Everything downstream of the outcome brief is rigorous, which makes the evidence behind the brief the weakest remaining link. Internal consensus is to customer truth what consulting is to governing.

The failure mode is the brief written from the house. Every seat on the guild was consulted except a customer. The brief reads confidently, cites nothing anyone could be shown, and the fleet builds it beautifully. The tell is an outcome accepted

for weeks whose usage numbers never move. The correction is the citation rule, held at brief authoring the same way the harness is held at acceptance.

This stream carries its own health test beside the two above: the customer evidence behind each active brief stays dated and recent. When the dates age, the network has started guessing.

The New Layer

Every prior framework assumed domain knowledge was a given.

In Scrum, the Product Owner is expected to hold domain knowledge but the framework says nothing about where it comes from or how it is maintained across a portfolio. In SAFe, domain expertise is distributed across roles but there is no governing structure that owns correctness as a distinct accountability. In traditional project management, subject matter experts are consulted on request, and their knowledge lives in the documents they were asked to review.

These were reasonable designs for their era. When execution was the constraint, the pace of delivery gave domain experts time to be found. A two-week sprint left room for a three-day wait while the right person was located and briefed. A quarterly release cycle left room for a review phase where compliance could read what was built and flag what was wrong.

> *When AI makes execution cheap, domain context and timely judgment become the scarce resources. Scarce resources need structure.*

When the daily cycle replaces the two-week sprint, a three-day wait for domain knowledge is not a delay. It is a structural failure. When continuous delivery replaces the quarterly release, a review phase at the end is not governance. It is an autopsy.

The Domain Knowledge Network exists because the operating model requires domain knowledge to arrive before the build starts, to be encoded into artifacts that outlive the conversation, and to be maintained by a structure that learns from the field. No prior framework required any of these three properties, because no earlier operating model ran at a tempo where their absence was fatal.

Three properties distinguish this layer from anything that preceded it.

It is governed, not informal. Members are named, funded, and accountable. Their contributions are tracked, their availability is planned, and their impact is measured. This is not a community of practice or a center of excellence. It is a governing structure with specific obligations.

It encodes, not merely advises. The output of every contribution is an executable check or a persistent context entry, not a recommendation in a document. The expert's judgment runs on every push after they leave the room.

It compounds. Each contribution reduces the need for the next one. A network that runs for twelve months should require less expert time per outcome than a network that ran for three, because the harness and the knowledge graph carry forward everything that was encoded. A network where expert time per outcome is flat has failed the encoding discipline and must correct it.

Failure Mode: Network Becomes a Help Desk

The most common way the network degrades is from a governing structure to a help desk. The tell is specific: experts are pulled in reactively after teams stall, rather than injecting knowledge before the build starts.

When this happens, the push model has collapsed into a pull model. Teams build without domain context, hit a question they cannot answer, search for the right expert, wait for availability, and receive an answer that arrives too late to prevent the rework. The expert's contribution is reactive troubleshooting rather than proactive governance. They are fixing what went wrong rather than preventing it from going wrong.

The correction is to restore the push model driven by daily planning. The signal already exists: yesterday's daily planning names the expert tomorrow's spec needs. The network reads that signal and connects the expert before the day starts. If the signal is not being generated, the problem is in daily planning, not in the network. If the signal is generated but not read, the problem is in the coordination mechanism.

> *The tell is unmistakable: experts pulled reactively after teams stall. The correction is always the same: restore the push. The signal comes from daily planning. The dispatch happens before the build starts.*

A second diagnostic confirms the diagnosis. Track whether expert contributions happen before the build (injection) or after the build stalls (troubleshooting). In a healthy network, 80% or more of expert time is injection. When troubleshooting exceeds 20%, the push model is failing and the network is becoming the help desk it was designed to replace.

The Expert on Request, the Embedded Analyst, and the Review Phase

The Domain Knowledge Network replaces three structures that exist in most organizations.

The subject matter expert on request. In traditional project management, subject matter experts are consulted when a team raises a question. Their availability is unplanned, their contribution is unstructured, and their knowledge lives in the documents they were asked to review. The network replaces this with planned, funded, encoded contributions.

The embedded analyst. Some organizations assign business analysts to standing teams for the life of a project. This creates depth for one team and starvation for every other. The network replaces dedication with floating, which serves the portfolio rather than a single team.

The review phase. Compliance review, policy review, and domain review typically happen after the build, creating a queue and a rework cycle. The network moves domain knowledge to the front of the build, eliminating the review queue for everything the harness can check.

How the Network Compounds

The Domain Knowledge Network depends on and supports several other elements of the operating model.

Knowledge investment. The network is the organizational structure that executes the knowledge investment strategy. Knowledge investment describes what to invest in. The network describes who invests and how.

Decision ownership. The Decision Owner is drawn from the same business guild as the network and is the only exception to the floating rule. The network exists for reach beyond what the owner carries.

The harness. Encoding is the step that connects the network to the harness. Expert criteria become checks that run on every push. Without the harness, encoding has no destination and the network cannot compound.

Does your organization have a structure that owns correctness?

30 minutes to discuss the Domain Knowledge Network for your portfolio. No pitch.

Book a call | singularics.ai

SINGULARICS

We help large organizations close the intent gap between strategy and execution so that when they accelerate with AI, they accelerate in the right direction.

singularics.ai | Book a conversation