The Engineering Behind the AI That Gets Smarter Every Day

Why generic AI fails technicians — and how Rudy is engineered to know the trade, the equipment, and the lingo

White Paper v5.1

Nick Haschka Founder, AskRudy — CEO, OnPoint Generators, Inc.

July 2026

Executive Summary

Field service industries face an accelerating knowledge crisis. Experienced technicians are retiring faster than they can be replaced. The training pipeline is structurally broken — the people best qualified to teach are the ones organizations can least afford to pull off revenue-generating work.

The market's reflex answer — point technicians at ChatGPT — fails at the point of application. Ask a general-purpose assistant for a fault code on a specific controller and you get a fluent, confident answer that may be subtly wrong: hallucinated from adjacent training data, blind to the model year, the trade's terminology, and the safety context. For a homework question, that's an inconvenience. For a technician working on live equipment, it's a liability.

Rudy is engineered to be fundamentally different for this job. **It knows the context** — the conversation, the technician's fleet, and the organization's own documents. **It knows the trade** — the terminology, the hazards, and how experienced techs actually phrase problems. **It knows the equipment** — manufacturer, model, and vintage, with retrieval hard-scoped so a Cummins question is never answered from a Caterpillar manual. **And it knows the lingo** — the jargon, abbreviations, and aliases technicians actually use, learned from real usage. Every answer is grounded in ingested documentation — including the spec tables and wiring diagrams buried inside manuals, which the pipeline extracts as first-class searchable content — and cited to the page, so the technician can verify rather than trust.

Rudy is a production platform purpose-built to solve this problem. What's deployed and working today: three-source hybrid retrieval combining vector search, full-text keyword search, and knowledge graph traversal; an Instructed Retriever that generates multi-step search plans rather than single-query lookups; iterative retrieval with automatic gap detection and query reformulation; query-focused context compression; a self-healing knowledge loop that detects gaps, researches autonomously, and ingests results; 16 designed self-reinforcing learning loops (11 operational today, the rest activating as usage grows); a full safety architecture covering 8 hazard categories with per-tenant calibration; and a multi-tenant platform with a 3,000+ document acquisition pipeline — 1,300+ manuals indexed and searchable today — across 127 manufacturers and 187 equipment models. A full distribution and community engine is deployed: Rudy Cards for saving and sharing research, coverage transparency with tiered research depth, peer validation, equipment expertise badges, team and community research libraries, public card pages with SEO, and Stripe billing across consumer and B2B team tiers. Conversation history is now permanent and fully searchable — durable in Postgres, equipment-scoped, and retrievable indefinitely rather than expiring from a cache. And the platform has begun exposing its retrieval and diagnostic engine as a service-token-authenticated API: the first embedded integration is live, letting a generator-monitoring product resolve equipment profiles and fault-code diagnoses at device-commissioning time. Production usage with real technicians on askrudy.io has begun, and the first real-world conversations are already driving measurable quality improvements.

What makes this durable rather than merely feature-complete: four compounding layers that strengthen every day the platform operates. Individual compounding — the more a technician uses Rudy, the more Rudy understands their equipment context and expertise level. Community compounding — every correction weighted by authority, every implicit feedback signal, every tribal knowledge tip makes the knowledge base more accurate without requiring explicit effort. Content compounding — user queries surface gaps, the self-healing loop fills them autonomously, and the flywheel accelerates. Cross-trade compounding — diagnostic patterns, safety frameworks, and learning infrastructure transfer structurally to adjacent verticals, so each new trade inherits from every prior one.

The architecture is domain-agnostic. Power generation is the proving ground. Diesel mechanics, marine, heavy equipment, forklifts, pumps, cranes, refrigeration, industrial electrical, elevator, and HVAC are the expansion sequence — ordered by knowledge domain overlap so each trade inherits from the last. The result is a platform whose accuracy, trust, and coverage compound with every month of operation.

Part I: The Problem

1. Expertise Is Disappearing

When a 30-year power generation technician retires, they take with them a form of knowledge that no documentation system captures. Pattern recognition developed over thousands of service calls. Equipment-specific quirks that never made it into any manual. Diagnostic shortcuts that turn four-hour jobs into one-hour jobs. Safety insights learned from near-misses that were never written down because the experienced tech knew to avoid the situation in the first place.

The scale of this retirement wave is not a theoretical concern:

- The IEA reports 2.4 energy workers nearing retirement for every new entrant under 25
- Goldman Sachs estimates 750,000+ new power industry workers will be needed by 2030
- The Department of Labor projects nearly half the current power workforce will retire within a decade

Knowledge is fragmented across manufacturer manuals (often outdated), service bulletins (scattered across dealer portals), regulatory codes (paywalled), and individual experience (leaves when the person leaves). No existing system captures the intersection of all four.

2. The Training Pipeline Is Structurally Broken

The expertise crisis would be manageable if training worked. It doesn't — not at scale, not at the margins required.

The fundamental problem is economic. The best trainer in any field service organization is the master technician. That person costs $85 to $120 per hour in loaded labor. Sending them to ride along on a routine call that a junior tech should be able to handle alone destroys the economics of that truck roll. So organizations don't do it. Juniors learn through trial and error, callbacks accumulate, and customer satisfaction erodes.

The problem compounds in several directions:

Resentment. Senior technicians signed up for complex diagnostic work — the problems that require genuine expertise. Babysitting a junior through a 40-point load bank test is not what they trained for. Organizations that assign training duty to their best people often lose them to competitors who don't.

Transfer failure. Classroom training and online modules teach procedures in the abstract. Field competence requires doing, with someone experienced watching and correcting. A junior who can recite the steps for a transfer switch commissioning from a slide deck may freeze in front of an unfamiliar panel. The knowledge doesn't transfer until it's applied under observation.

Availability mismatch. Emergencies don't schedule themselves. A junior tech facing an unexpected fault condition at 2 AM needs the master tech's knowledge, not a call center. The master tech is unavailable, or unavailable for this level of question, or unavailable at this hour.

This is why Rudy matters at a structural level beyond "better search." Rudy is the senior technician riding along on every job, available 24/7, infinitely patient with repeated questions, never resentful of the interruption, and costing nothing incremental per truck roll. It doesn't replace the master technician. It makes the master technician's knowledge available at the moment and location where the junior technician needs it.

3. Why Generic AI Fails Technicians

The market's first response to this problem was to point field service organizations at general-purpose AI assistants — ChatGPT and its peers. These tools are impressive in consumer contexts. They fail in technical field service for reasons that are architectural, not superficial:

Single-pass retrieval. Standard RAG (Retrieval-Augmented Generation) systems embed the query, find similar documents, and synthesize an answer in a single pass. Real diagnostic questions don't work this way. "The QSB7 won't start after a coolant flush — what did we miss?" requires identifying the symptom, correlating with known failure patterns on that engine family, checking the maintenance procedure for common errors, and cross-referencing the coolant system's interaction with the air bleeding procedure. A single retrieval pass can't follow that chain.

Generic embeddings. When a technician queries "DSE 8610 Alarm 1437," the embedding must find an exact match against that controller's fault code table — not a semantically similar passage about alarm conditions in general. Generic models trained on internet text treat trade-specific alphanumeric identifiers as noise.

No expertise weighting. Standard RAG treats a manufacturer service bulletin and a Reddit post as equally authoritative. A field service knowledge system must weight sources by provenance, recency, and the authority of contributors.

No safety awareness. A query about replacing a starter motor on a large genset involves working adjacent to a battery bank capable of delivering thousands of amperes. Generic AI doesn't know this. It answers the mechanical question and omits the electrical safety context entirely. In a safety-critical domain, that omission is not a quality issue. It is a liability.

Hallucination in safety-critical domains. Generic AI systems are calibrated to provide confident, helpful answers. In field service, a confidently wrong torque specification or an incorrectly recalled wiring diagram can cause equipment damage, downtime, or injury. The error cost is asymmetric — a "I don't have enough information" answer is recoverable; a confident wrong answer may not be.

Blind to what's buried in manuals. The answer a technician needs is often not prose at all — it's a row in a specification table, a callout on a wiring diagram, a maintenance-interval matrix. Generic assistants either never saw that content or flattened it into noise during training. A system that can't read the tables and diagrams inside OEM manuals is working from a fraction of the real documentation.

None of these are prompt-engineering problems. They are architecture problems, and solving them is what the rest of this paper describes: a system that knows the context, knows the trade, knows the equipment, and knows the lingo — because every layer of it was built for technicians rather than adapted from a consumer chatbot.

Part II: What's Built

Overview

Rudy's production system is not a wrapper around a general-purpose model. It is a purpose-built retrieval and reasoning platform with a system of self-reinforcing learning loops (11 of 16 designed loops operating today), a

self-healing knowledge pipeline, and a safety architecture that treats hazard detection as an architectural constraint rather than a feature.

The following subsections describe each major component. The emphasis throughout is engineering depth — not just what the system does, but why each piece exists and how the pieces reinforce each other.

Built for Technicians: Context, Trade, Equipment, Lingo

Four properties separate a tool built for technicians from a chatbot with a system prompt. Each maps to specific machinery described in the sections that follow:

Knows the context. Conversations are permanent, searchable, and equipment-scoped. A technician's fleet accumulates from usage and is injected into every answer for disambiguation. Tenant documents — the organization's own SOPs and site manuals — outrank everything else in retrieval.

Knows the trade. An 80+ term domain dictionary, a safety architecture spanning 8 hazard categories, and an authority system that weights corrections by demonstrated expertise. The system understands that a load bank test and a transfer switch commissioning are different jobs with different hazards.

Knows the equipment. A knowledge graph of 120,000+ relationships connects manufacturers, models, fault codes, symptoms, causes, and resolutions. Manufacturer-aware retrieval boosts matching-equipment content 2x and penalizes mismatches to 0.4x — a Cummins question is never answered from a semantically similar Caterpillar page. Persona calibration infers what each technician already knows and adjusts answer depth to match.

Knows the lingo. Jargon, abbreviations, and equipment aliases are learned from real query patterns. A 27-cluster synonym expansion covers the terms technicians use interchangeably. "Genset won't turn over," "engine fails to crank," and a bare fault-code string all reach the same evidence.

Three-Source Hybrid Retrieval

Rudy retrieves evidence from three complementary sources simultaneously, each capturing a different dimension of relevance that the others cannot:

Vector Search (Qdrant, Gemini Embedding-001, 768 dimensions) — Dense embeddings capture semantic meaning independent of exact terminology. The system understands that "genset won't turn over" relates to "engine fails to crank" without keyword overlap. Embeddings are cached in Redis with a 7-day TTL to eliminate redundant computation. Embeddings are batch-processed with automatic retry logic (3 attempts, exponential backoff).

Keyword Search (PostgreSQL BM25) — Full-text search captures exact terminology matches that semantic similarity misses. When a technician queries a specific part number, fault code, or model designation, lexical matching finds it directly. "DSE 8610 Alarm 1437" must match the fault table entry for that exact alarm on that exact controller — semantic proximity is insufficient.

Knowledge Graph Traversal (PostgreSQL recursive CTEs) — Structural relationships that neither vector nor keyword search can represent: equipment hierarchies (manufacturer → model → system → component → part), diagnostic chains (fault code → symptoms → root causes → resolutions), and cross-document entity relationships. The graph contains 120,000+ relationships with deterministic entity UUIDs and cycle-safe traversal up to 4 hops deep.

Reciprocal Rank Fusion (RRF) merges results from all three sources into a unified ranking without requiring comparable scores across heterogeneous retrieval systems. Graph results receive a **1.5x boost** during rank fusion, reflecting the high precision of structural traversal for technical queries. All three sources are configured as non-fatal: if any single source fails, retrieval continues with the remaining sources.

Manufacturer-aware boosting further protects answer integrity. Chunks whose manufacturer matches the detected equipment receive a **2x boost**, while mismatched chunks are demoted with a **0.4x penalty** — a calibration refined directly from tracing real production conversations. The result is that a Cummins query is not answered from a topically similar Caterpillar or Kohler document, even when the wrong-manufacturer chunk is semantically close. When the pre-boost top retrieval score is low (below 0.5), the synthesis prompt receives an explicit caveat instructing the model to separate document-sourced facts from general knowledge and to flag any cross-manufacturer mismatch.

Full-Depth Ingestion: The Data Buried in Manuals

Most of what a technician actually needs from a 400-page OEM manual is not prose. It is a row in a specification table, a wiring diagram callout, a maintenance-interval matrix, a fault-indicator legend. Ingestion pipelines that only extract paragraphs discard exactly the content that answers real field questions. Rudy's pipeline treats tables and graphics as first-class content:

Tables with preserved context. Table chunks are embedded with their section breadcrumbs and the preceding paragraph injected into the embedding text. A specification row for oil capacity carries the knowledge that it belongs to "Engine Specifications — KTA19-G3," so a query like "what's the oil capacity of a KTA19" retrieves the right row of the right table — not a lexically similar row from a different model's manual.

Diagram and image understanding. A vision pipeline (Gemini multimodal) analyzes flowcharts, wiring diagrams, spec tables, and equipment photos, converting graphical content into searchable, citable text — and as of July 2026 this analysis has been applied across the full indexed corpus, so every indexed manual's graphical content is retrievable evidence. The fault-indicator table printed as an image on page 39 of a service manual becomes an answer with a page-level citation.

Verbatim spec extraction. For manual-type documents, a chunk-aware extraction pass reads section headings and collects specification text directly from the chunked source — actual torque values, fluid capacities, and clearances extracted verbatim rather than paraphrased by a model. Domain-specific extraction profiles define which fields matter per document type (manual vs. spec sheet vs. service bulletin).

Quality measured, not assumed. A six-hypothesis automated test suite (H1–H6) measures context preservation, insight capture, data-object detection, page coverage, knowledge synthesis, and extraction accuracy against ground truth on every pipeline change, with week-over-week regression detection and automated remediation.

The result is that Rudy's answers can cite the buried content — the table row, the diagram callout — that a text-only RAG system never ingested and a general-purpose assistant never saw. Not for a sample of documents: for the entire indexed library.

The Instructed Retriever

Standard RAG converts a user question into a single embedding lookup. The Instructed Retriever treats retrieval as a multi-step reasoning problem.

When a query arrives, the system generates a **search plan** — a structured strategy that decomposes complex questions into sub-queries, specifies which retrieval sources to emphasize for each sub-query, identifies entity references requiring resolution, and defines what constitutes sufficient evidence for a confident answer.

A query like "What's the oil change procedure for a Cummins QSB7 and what's different about the marine variant?" generates a plan that decomposes into two sub-queries (standard procedure + variant differences), routes each to the appropriate combination of retrieval sources, and defines a merge strategy that presents the standard procedure with marine-specific annotations.

The search plan includes metadata filters — manufacturer, document types, keyword terms — that are wired into both Qdrant vector search and PostgreSQL BM25 keyword search. Reranking is **intent-aware**: a `SPEC_LOOKUP` query emphasizes numerical values; a `TROUBLESHOOT` query emphasizes fault codes and resolution steps. This per-intent scoring transforms reranking from a generic relevance operation into a domain-appropriate judgment.

The planning layer transforms retrieval from statistical similarity into a reasoning process that mirrors how an experienced technician would search for information across a library of sources.

Iterative Retrieval with Gap Detection

When the first retrieval pass doesn't yield sufficient evidence, Rudy automatically detects the gap type and reformulates:

Gap Type	Condition	Response
`vector_empty`	No vector search results	Broaden semantic query, remove filters
`weak_matches`	Low similarity scores	Reformulate with alternative terminology
`graph_empty`	No graph traversal results	Expand search radius, try alternate entry points
`insufficient_evidence`	Low confidence in available evidence	Decompose query, search for supporting sub-topics
`filter_over_constrained`	Too-restrictive metadata filters	Fast path: drop manufacturer filter, retry immediately

The loop runs up to 3 rounds within a 15-second total timeout. The `filter_over_constrained` fast path handles the common case where a technician asks about a specific manufacturer but the relevant cross-reference appears in a multi-manufacturer technical document. Context is preserved across rounds — evidence accumulates rather than restarting.

Gap analysis reads a structured **retrieval manifest** logged per query: source counts, similarity scores, merge events, and boost events. This manifest is the prerequisite for gap detection — the loop is not guessing whether retrieval failed but reading structured diagnostics that describe exactly why.

Context Compression (KARL)

Real-world technical queries often retrieve more evidence than can be synthesized in a single pass. The Knowledge-Aware Retrieval with Lossy compression (KARL) system activates when more than 10 chunks are retrieved, compressing them into a dense, query-focused narrative using Gemini Flash Lite. The maximum synthesis chunk count is 25.

The compression pipeline preserves citation metadata through the compression process. Compressed evidence maintains full traceability to source documents, sections, and page numbers — a regulatory requirement in environments governed by OSHA, NFPA, and EPA, where demonstrating the provenance of every piece of guidance is not optional.

Self-Healing Knowledge Loop

The self-healing loop is one of the more architecturally unusual components of the system. It closes the feedback loop between quality evaluation and knowledge acquisition automatically, without human intervention:

1. The LLM-as-Judge quality pipeline runs against 96 golden questions organized across 4 complexity tiers
2. Answers that receive an F grade are flagged as knowledge gaps
3. The gap triggers an autonomous research job: Gemini with `google_search` grounding researches the topic, generating a synthesis document
4. The synthesis document is ingested into the knowledge base with `synthesized_research` authority tier (0.75)
5. A regression quarantine verifies that the new content doesn't degrade scores on previously passing questions
6. The `research_topics` table prevents duplicate research on the same gap

Since activation, 30+ research documents have been auto-generated and ingested. The self-healing loop runs automatically after every golden test cycle, converting F-grade gaps to research targets without any manual trigger.

A stale job reaper complements this: jobs stuck in processing for more than 30 minutes are automatically reset; permanent failure is declared after 3 crashes. The ingestion pipeline heals itself.

Self-Reinforcing Learning Loops

Most AI systems are static after deployment — they answer the same way on day 1,000 as on day 1. Rudy is designed so that every interaction makes the next interaction better, without retraining and without human intervention. Sixteen loops are designed into the platform; eleven operate continuously today across six domains, with the remaining five activating as usage volume grows:

Retrieval gets sharper. Every citation a technician clicks is a relevance signal. Chunks with consistently high click-through rates for related queries receive ranking boosts. The system converges toward surfacing the most useful content for each query type — automatically, from usage.

Expert knowledge rises to the top. A 5-tier authority system tracks who contributes reliable corrections and who doesn't. A master tech's correction carries more weight than an apprentice's guess. Approved corrections boost authority; rejected ones penalize it. The knowledge base converges toward truth, not toward volume.

Safety calibrates itself. Too many warnings cause alert fatigue; too few miss genuine hazards. A proportional controller adjusts per-tenant safety sensitivity based on how technicians respond to warnings. Each organization converges on the right balance for their specific equipment and risk profile.

The system learns your language. Jargon, abbreviations, and equipment aliases are learned from real query patterns. When technicians consistently use shorthand the system doesn't recognize, it learns the mapping. The domain dictionary grows from usage.

Rudy adapts to each technician. Query patterns reveal skill level — the system infers whether to give apprentice-level detail or expert-level brevity. Equipment auto-detection learns which makes and models each technician works on regularly.

Quality improves invisibly. Ingestion quality scoring catches and reprocesses low-quality document chunks. Answer synthesis analysis measures satisfaction by intent type and tunes prompts accordingly. Community pattern detection surfaces emerging knowledge gaps before they accumulate into clusters of bad answers.

Feedback flows without effort. Technicians rarely click thumbs-up/thumbs-down — they're busy. So the system detects behavioral signals from follow-up messages: corrections ("actually it's..."), frustration (repeated questions, all-caps), confusion ("what do you mean"), and satisfaction ("that worked, thanks"). These signals flow continuously at zero added cost, creating a feedback stream that requires no explicit action from the technician.

The critical property of these loops is that they compound. Each loop generates data that other loops consume. Retrieval improvements surface better content, which generates more positive feedback, which strengthens authority signals, which improves the quality of corrections, which makes retrieval better. The system doesn't just learn — it learns faster the more it has already learned.

Safety as Architecture

Safety is not a layer applied after the core retrieval system works. It is an architectural constraint that shapes every component.

Safety analysis runs on every query and every retrieved chunk using pattern-matched detection across 8 hazard categories:

Category	Default Severity
Electrical (high voltage, arc flash, energized equipment, exposed conductor)	Danger
Mechanical (rotating equipment, pinch points, crush hazard, entanglement)	Danger
Fire (flammable materials, hot work, explosive atmosphere, flash point)	Danger
Pressure (pressure vessels, overpressure, rupture, safety valve)	Danger
Lockout/Tagout (LOTO, energy isolation, zero energy, stored energy release)	Danger
Chemical (hazardous materials, toxic, corrosive, inhalation hazard)	Warning
Environmental (spills, hazardous waste, groundwater contamination)	Warning
PPE (personal protective equipment, fall protection, respirator)	Caution

Detection uses pre-compiled regex with case-insensitive word boundary matching for real-time performance. The risk level is calculated from the combination of detected hazards: two or more Danger-level detections produce Critical overall risk; one Danger produces High risk.

Safety warnings are mandatory. A query about "replacing the starter motor" that involves working near a battery bank will include electrical safety warnings even though the question was mechanical. The hazard is injected regardless of whether the technician asked about it.

Per-tenant calibration uses a proportional controller: if technicians are dismissing more than 30% of safety warnings (alert fatigue), sensitivity decreases. If dismissal rate is below 30%, sensitivity increases. The adjustment threshold is 0.05 — changes smaller than 5% are not applied, preventing oscillation. Calibration requires a minimum of 50 interactions before activating.

Every safety-relevant interaction is logged: query content, user identity, tenant scope, detected categories and severities, warning acknowledgment or dismissal, and full citation chains. This audit trail supports OSHA, NFPA 110, EPA, and other regulatory frameworks where demonstrating the accuracy and provenance of safety information is a legal requirement.

Relational Architecture

Technical accuracy is necessary but not sufficient. A system that gives correct answers while making technicians feel stupid for asking will not be used. This is not a UX polish concern — it is an adoption-critical architectural decision.

Rudy's relational design is codified in a **relationship mandate** applied to every synthesis prompt:

- **Never make a technician feel stupid for asking anything.** The senior tech in your pocket who never judges.
- **Earn trust by being right, not by claiming authority.** Every answer is cited. Confidence is earned from outcomes, not asserted.
- **Say "I don't know" plainly.** Never guess specifications, torque values, or wiring configurations. Uncertainty is stated directly.
- **Correct without condescension.** When a technician's premise is wrong, acknowledge their reasoning before redirecting.
- **Give credit for good instincts.** When a technician is on the right track, say so explicitly.
- **Match their energy.** Terse question, terse answer. Detailed question, detailed answer. Per-user `detailLevel` preferences (brief/standard/detailed) persist across sessions.
- **Leave every conversation open.** Never close a thread. Always signal availability for follow-up.
- **Own mistakes when corrected.** When a technician provides a correction, acknowledge it immediately and update.

This relational design is not subjective — it is **measured**. A dual-track quality evaluation system scores every interaction on both technical accuracy (6 dimensions) and relational quality (10 dimensions, with non-judgmental safety weighted 2x). The relational judge measures trust-building, energy matching, humility, and whether the interaction leaves the technician more likely to return.

A lifecycle simulation framework tests this across 20 technician personas over compressed 60-day lifecycles, measuring whether Rudy's relational approach produces daily retention scores of "strong_yes" or "probably" rather than "at_risk." Personalization delta testing compares paired accounts (one fresh, one with history) to verify that the relationship deepens with use.

This relational calibration is the product of a purpose-built measurement framework — thousands of simulated lifecycle interactions scored against a 10-dimension relational judge — and it sharpens with every evaluation cycle.

LLM-as-Judge Quality Evaluation

Automated quality evaluation runs continuously using a dual-track LLM-as-Judge pipeline.

Technical Judge — scores every answer across six dimensions:

Dimension	What It Measures
Factual Accuracy	Are the facts in the answer correct?
Safety Correctness	Are safety warnings appropriate and accurate?
Hallucination	Does the answer fabricate claims not supported by evidence?
Context Adherence	Does the answer stay grounded in the retrieved context?
Completeness	Does the answer cover all aspects of the question?
Actionability	Can a technician act on this answer in the field?

Hallucination detection is weighted most heavily because in a safety-critical domain, a confidently wrong torque specification or an incorrectly recalled wiring diagram can cause equipment damage or injury. The system is calibrated to strongly prefer "I don't have enough information" over a confident wrong answer.

Relational Judge — scores every interaction across 10 dimensions measuring how the technician *feels* about the interaction, not just whether the facts are correct. Non-judgmental safety (did the response make the technician feel stupid?) is weighted 2x because a technically perfect answer that humiliates the user destroys retention. Additional dimensions include trust-building, energy matching, humility, correction gentleness, credit-giving, and conversational openness.

Lifecycle Judge — evaluates daily retention meta-scores (strong_yes / probably / uncertain / at_risk) across 20 simulated technician personas operating through compressed 60-day lifecycles. Personalization delta testing samples paired accounts on days 1, 15, 30, 45, and 60 to measure whether the system actually improves for returning users versus fresh accounts. Adversarial injection tests safety-critical scenarios including attempts to extract dangerous information or bypass safety warnings.

The golden test suite contains 96 questions across 3 complexity tiers (S1 single-source, S2 two-source, ML multi-layer synthesis), including 15 cross-document questions that require synthesizing information from multiple sources. Multi-layer question generation uses Gemini 2.5 Pro; grading uses Flash for S1/S2 and Pro for ML. The tiered structure measures not just recall of individual document facts but synthesis capability across the full knowledge base.

Parallel General-Knowledge Fallback

A core design principle: Rudy must never give a worse answer than a base model. When RAG retrieval fails to surface the relevant chunk, the system should still answer correctly from base model knowledge rather than returning "not found."

The parallel fallback pipeline runs via `Promise.all()` — both the full RAG retrieval and a general-knowledge generation call launch simultaneously. The fallback adds zero latency because it completes while RAG retrieval is still running.

A second LLM call evaluates the fallback answer's accuracy, producing a confidence rating (HIGH, MEDIUM, or LOW) with reasoning. The synthesis prompt then merges all available knowledge with strict priority ordering:

1. **Tier 1 (highest):** Customer documentation — manufacturer manuals, spec sheets, bulletins
2. **Tier 2:** Graph knowledge — structured relationships from the knowledge graph
3. **Tier 3:** User-contributed knowledge — tribal knowledge tips, verified corrections
4. **Tier 4 (lowest):** General knowledge — base model answers with confidence indicators

The SSE streaming response includes `knowledgeSources` metadata showing which tiers contributed. When general knowledge is used, its confidence level is displayed so technicians can make informed judgments about verification.

Multi-Tenant Platform

Rudy operates as a fully configurable multi-tenant platform. Every database query, API call, cache key, and vector search is tenant-scoped. Data isolation is enforced at the infrastructure level — there is no code path through which one tenant's private documents could appear in another tenant's search results. A SOC 2 readiness blueprint — encryption standards, audit logging, access management, and change control — is defined and being implemented in preparation for enterprise procurement.

The knowledge architecture uses a two-tier model:

Foundational knowledge (shared) — A curated corpus of 239 cross-trade resources — DOE Fundamentals Handbooks, thermodynamics references, electrical theory, safety standards, and manufacturer-agnostic engineering materials — is available to all tenants automatically. This ensures that every new tenant has immediate value on day one, before any domain-specific content is uploaded. The foundational tenant's documents are included in vector search filters, keyword search, diagnostic graph queries, and knowledge lookups for all tenants.

Tenant-private knowledge (isolated) — All customer-uploaded documents, SOPs, site-specific manuals, and proprietary content is strictly tenant-scoped. A manufacturer's proprietary service bulletin uploaded by one tenant is never visible to another. The private layer builds on top of the foundational layer — retrieval merges results from both, with tenant-private content always ranked above foundational content.

This architecture directly enables the trade expansion strategy: launching a new vertical does not require building from zero. The foundational corpus covers thermodynamics, electrical theory, safety, and mechanical engineering principles that transfer to every trade. Domain-specific content — the Allison transmission manual for diesel trucks, the Carrier VRF documentation for HVAC — layers on top.

Adaptive AI-driven onboarding reduces tenant setup from days to minutes: Rudy crawls the tenant's website to extract business context, conducts a targeted AI interview to fill gaps, then auto-generates a complete configuration — system prompt, categories, safety rules, RAG tuning parameters, and starter questions.

Per-tenant configuration includes: credits, authority tiers, safety sensitivity levels, knowledge extraction results, feedback loop parameters, and all retrieval parameters. A tenant operating a fleet of Caterpillar generators gets different defaults, different safety calibration, and different query patterns than one operating mixed-manufacturer industrial equipment.

Embodied Knowledge Extraction

Beyond retrieving from documents, Rudy extracts structured knowledge from its corpus and builds it into a queryable layer:

Glossary Terms — Abbreviations with full forms and definitions, extracted automatically from each ingested document. When a technician uses "LOTO" in a query, the system expands to "Lockout/Tagout" and searches for both forms.

Equipment Entities — Canonical equipment names with manufacturer aliases. "CAT C15," "Caterpillar C15," and "C-15 ACERT" all map to the same engine. At query time, a question about any name variant finds content tagged with all variants.

Chunk-Aware Spec Extraction — For manual-type documents, a specialized extraction pipeline reads section headings from document chunks and collects specification text directly from the chunked content rather than relying on LLM generation. This produces higher-fidelity specifications — actual torque values, fluid capacities, and clearances are extracted verbatim from the source material rather than hallucinated by a model. Domain-specific extraction profiles define which fields to look for per document type (e.g., manual vs. spec sheet vs. service bulletin).

Extraction runs automatically when document ingestion completes (Pipeline B) and as a cross-document resolution batch (Pipeline C) that merges duplicate entities discovered across different documents.

Platform API & Embedded Integrations

Rudy's retrieval and diagnostic engine is not only an end-user application — it is exposed as a service-token-authenticated API that other products can embed. The first integration is live in production: a generator-monitoring product (Pulse) calls Rudy during device commissioning.

Method	Endpoint	Purpose
GET	`/api/pulse/equipment/:make/:model`	Resolve an equipment profile by manufacturer and model
POST	`/api/pulse/diagnose`	Diagnose a fault code, returning ranked probable causes with confidence and suggested remediation

When the requested equipment isn't yet covered, the lookup returns a not-matched indicator rather than an error — the monitoring product keeps working while Rudy's knowledge base expands. Authentication uses a dedicated service token validated in constant time.

This is strategically significant. It turns Rudy from a destination into infrastructure: the same retrieval, knowledge-graph, and fault-diagnosis engine that powers the technician-facing chat can be embedded inside hardware, monitoring platforms, and OEM tools. Each integration becomes a new distribution channel that feeds queries — and therefore learning signal and gap detection — back into the same compounding engine. The current endpoints accept make/model identifiers and fault codes (not raw sensor telemetry), but they establish the stable API contract through which richer integrations can be built as described in Part VI.

Additional Production Capabilities

The following capabilities are deployed and working, listed for completeness:

- 55+ admin API endpoints covering stats, corrections, learning, discovery, authority, knowledge, users, query analytics, and synthetic testing
- Customer success analytics with tenant-scoped metrics including active users, session counts, latency percentiles (avg/p50/p95), and satisfaction rates
- Query normalization: 80+ term domain dictionary, 27-cluster synonym expansion for BM25, equipment entity extraction for garbled queries, optional Gemini Flash Lite rewrite with truncated-query completion
- Confidence-gated validation routing uncertain queries through enhanced reasoning with re-retrieval
- Conversational follow-up with a permanent message store: every turn is dual-written to a Postgres `conversation_messages` table (the permanent record) and a Redis hot cache (30-day TTL), so history is available indefinitely rather than expiring after a week. Follow-up clarifications (e.g. a bare "generac" or a fault-code string) bypass the readiness gate and are treated as contextual refinements of the active conversation
- Photo upload with client-side compression, GCS signed URL, and Gemini multimodal analysis
- Document acquisition pipeline discovering PDFs, downloading to GCS, and queuing for ingestion
- Query disambiguation: detects ambiguous queries and sends clarification hints while continuing to answer
- Async response groundedness evaluator: post-synthesis background verification that answer claims are grounded in retrieved chunks
- Rudy Cards: save answers as shareable research cards, My Library with search/equipment filters, public card pages at `/c/{slug}` with JSON-LD and OG image generation
- Coverage transparency: signal bars (strong/moderate/thin) on every answer, expandable Research Summary card
- Tiered research depth: Go Deeper (2 credits, expanded internal search) and Ludicrous Mode (5 credits, full external web research + cross-manufacturer analysis + downloadable PDF reports)
- Team Library: share research cards within tenant, admin pinning, contributor attribution with authority badges

• Community Research Library: cross-tenant anonymized sharing via Gemini Flash Lite PII stripping, peer validation ("This helped me"), equipment expertise badges (Familiar→Expert per manufacturer)

• B2B Team Expertise Dashboard: equipment expertise map, knowledge contributors ranking, team research library, knowledge gap analysis

• Growth infrastructure: MDX blog pipeline, interactive demo widget, ad landing pages, ROI calculator, conversion tracking, sitemap/robots/llms.txt, admin growth dashboard

• Stripe billing: graduated free tier (25 free answers, then 5/day), $9/mo unlimited consumer, $15/seat/mo B2B teams ($13 annual) with 14-day trial, billing groups with invite-code onboarding

• Chat history experience: full-text search across past conversations (Postgres `tsvector` + `websearch_to_tsquery` with weighted ranking), equipment-scoped filtering with equipment-name chips, pin/archive, message snippets and counts, infinite scroll, keyboard navigation, inline rename, Gemini auto-title, soft-delete semantics, and IDOR-hardened conversation resolution

• Smart PDF viewer: in-app manual reader with virtualized page rendering, mobile single-finger swipe-to-turn with page-turn animation, pinch-zoom, and RAG citation deep-linking — "a manual that understands itself"

• Role-aware synthesis: the system distinguishes dealers, field technicians, and plant operators and their differing access levels (fleet portals, remote configuration), and proactively researches unknown fault codes on the web rather than reporting them as undocumented

• Pipeline recovery: Wayback Machine fallback for 403 URLs, automatic retry with exponential backoff, auto-reject for stale failures

Content Scale (June 2026)

Metric	Value
Documents in acquisition pipeline	3,000+
Documents indexed and searchable	1,300+
Manufacturers covered	127+
Equipment models	187
Curated foundational resources	239
Searchable chunks	90,000+
Knowledge relationships	120,000+

Query Processing Pipeline

```
Technician Query
|
v
+-------------+
| Intent | <- Classify query type, extract entities
| Analysis |
+------+------+
|
```

```
v
+-------------+
| Search Plan | <- Instructed Retriever generates multi-step plan
| Generation | <- Entity alias expansion from knowledge graph
+------+------+
|
v
+-------------------------------------------------+
| Three-Source Parallel Retrieval |
| +-----------+ +-----------+ +-----------+ |
| | Vector | | Keyword | | Graph | |
| | (Qdrant) | | (BM25) | | (PG CTEs) | |
| +-----+-----+ +-----+-----+ +-----+-----+ |
| +---------------+-----------------+ |
| v |
| Reciprocal Rank Fusion |
| (Graph results: 1.5x boost) |
+------------------------+------------------------+
|
v
+-------------+
| Retrieval | <- Manifest: source counts, scores,
```

Part III: The Compounding Engine

This is the mechanism that converts a good product into a durable business. The engineering described in Part II can be rebuilt with time and talent. The compounding engine has to be grown — it requires users, and users generate the data that makes the next user's experience better, which attracts more users.

As of mid-2026, production usage with real technicians on askrudy.io has begun, and the first real-world conversations are already feeding the engine. The manufacturer-contamination guard, the low-confidence caveat, the follow-up readiness bypass, and proactive fault-code research shipped in June 2026 were each derived directly from tracing real user sessions — not from synthetic tests. The flywheel has started to turn: real usage surfaces real failure modes, which drive real fixes, which improve the experience for the next user.

Four layers operate simultaneously:

Layer 1: Individual Compounding

The more a technician uses Rudy, the more Rudy understands them specifically.

Persona calibration infers expertise level from question phrasing and behavioral patterns. A technician who asks "what does Alarm 1437 mean on a DSE 8610" is at a different competence level than one who asks "the DSE is showing 1437 but the coolant temp looks normal — is there a sensor calibration drift known on QSB7s?" The system infers the difference and calibrates response detail accordingly. Over time, this calibration becomes

increasingly accurate.

Equipment auto-detection tracks what a technician works on across sessions. The 24-hour staleness cutoff on auto-loaded equipment context means the system is always working with relevant fleet context, not stale data from a previous engagement. Cross-manufacturer poisoning is guarded against — if a tech switches from a Cummins to a Kohler job, the system detects the topic shift and resets context rather than contaminating the new query with the prior equipment's context.

Implicit feedback signal detection captures behavioral signals without any explicit rating effort. Message patterns indicating frustration (repeating the question, adding "I already tried that"), confusion (asking for clarification, requesting simpler language), or satisfaction (proceeding to follow-up questions rather than retrying) are detected via zero-cost regex and stored as continuous feedback signals.

Persistent conversation memory deepens this relationship. Every exchange is now stored permanently and made searchable — durable in Postgres rather than expiring from a cache — so a technician's entire history of questions, answers, and the equipment they pertained to accumulates into a personal, queryable knowledge record that grows more valuable the longer it is used.

The retention mechanism follows directly: Rudy's value to a specific technician accumulates — a calibrated persona, equipment context, implicit signals, and a searchable history of prior diagnoses that no fresh tool can offer on day one.

Layer 2: Community Compounding

The learning loops mean the platform improves autonomously from every interaction, not just from explicit feedback.

The contributor authority system ensures that quality converges toward truth rather than toward volume. A master technician's correction carries different weight than an apprentice's — the 5-tier points-based system reflects demonstrated expertise. Rejected corrections penalize authority; approved corrections build it. The knowledge base converges toward what the most experienced contributors validate, not toward what the most frequent contributors submit.

The safety calibration loop is a specific example of community compounding that has no analog in a static system. Different organizations have different risk tolerances, different equipment types, and different technician experience levels. A system with a single global safety sensitivity serves no organization well. The per-tenant proportional controller learns from actual dismissal patterns, converging on a calibration that reduces alert fatigue without sacrificing genuine hazard detection. That calibration is a learned artifact that took interactions to produce.

Citation click-through tracking creates a feedback signal that requires no explicit action from the technician. Every time someone clicks through to view a source document, that signal is recorded. Chunks with consistently high CTR for related queries are boosted in future rankings. The ranking system improves from usage without asking anyone to rate anything.

The community layer now extends beyond implicit signals. Rudy Cards allow technicians to save and share research as portable artifacts — shareable within teams (non-anonymized) or to the broader community (with LLM-driven PII stripping). Peer validation ("This helped me") creates explicit quality signals on shared research. Equipment expertise badges (Familiar through Expert, auto-computed per manufacturer from queries, contributions, and peer validations) make contributor credibility visible. A B2B Team Expertise Dashboard surfaces which technicians are specialists in which equipment families, turning implicit community knowledge into an organizational asset. Each of these mechanisms generates data that feeds back into retrieval ranking, authority scoring, and gap detection.

Retrieval can be built; this data can only be earned. Expert-weighted corrections, calibration signals, and citation engagement histories exist only as byproducts of operating the platform — they accumulate with usage and cannot be purchased or synthesized.

Layer 3: Content Compounding

The self-healing knowledge loop creates a flywheel between usage and content quality.

User queries are the detection mechanism. When a query against the golden test suite produces an F-grade answer, that is evidence of a knowledge gap. The gap triggers autonomous research — Gemini with `google_search` grounding produces a synthesis document. The document is ingested, regression-quarantined, and becomes part of the knowledge base. The next time a technician asks a similar question, the gap is filled.

But the flywheel is broader than the golden test suite. More users generate more query patterns. More query patterns surface more gap types. More gaps trigger more research. More research documents improve answer quality. Better answer quality attracts more users.

Document collection agents (in progress) extend this further: a Search Agent autonomously searches the web for relevant technical documents; a Browse-Along Agent collects materials from authenticated sites as technicians navigate them. Users with journeyman or higher authority can upload manuals directly. Every inbound document expands coverage, which improves answers for the entire user base.

The content flywheel has no natural ceiling. There is always more technical documentation to acquire, more failure patterns to document, more tribal knowledge to capture. The larger the knowledge base, the more comprehensive the gap detection, and the more precisely new research can target what's actually missing.

Layer 4: Cross-Trade Compounding

The architecture is domain-agnostic. The retrieval pipeline, safety framework, authority system, learning loops, quality evaluation, and multi-tenant platform all transfer structurally to adjacent trades.

The cross-trade compounding is not metaphorical — it is mechanical. A Cummins ISX in an emergency generator is the same engine as a Cummins ISX in a Peterbilt truck. Hydraulics failure modes on an excavator share root-cause patterns with hydraulics on a forklift and a crane. HVAC refrigerant circuit diagnostics follow the same logical structure as generator cooling system diagnostics. Safety frameworks for electrical hazards in power generation transfer directly to electrical hazards in HVAC and industrial equipment.

The expansion economics compound with each vertical:

- **First vertical (power generation):** Build everything — retrieval infrastructure, safety framework, authority system, learning loops, quality evaluation, multi-tenant platform, content for the specific trade
- **Second vertical (HVAC, diesel mechanics):** Reuse approximately 60% of infrastructure; seed with domain-specific content and terminology; calibrate safety patterns for the new domain
- **Third and fourth verticals:** Increasingly, the work is domain seeding rather than infrastructure construction
- **By the fifth vertical:** Mostly assembling from existing building blocks, with a narrowing slice of new work for domain-specific terminology and equipment hierarchies

Each vertical also contributes back to prior verticals. Diesel mechanics knowledge informs generator diagnostics. Industrial hydraulics knowledge informs heavy equipment maintenance. The cross-trade knowledge graph grows denser with each new domain, creating retrieval benefits that compound across the entire platform.

Each new vertical launches with everything the prior verticals taught the platform — by the fourth trade, a launch starts with four verticals' worth of compounded learning behind it.

Part IV: Why This Compounds Into Defensibility

Rudy's durability comes less from any single component than from how the systems — and the data they generate — reinforce each other. Honestly assessed, system by system:

Engineering Complexity

The individual components of Rudy's retrieval system are publicly documented. Vector search, BM25 keyword search, knowledge graph traversal, RRF rank fusion — these are known techniques with available tooling. Any competent ML engineering team can build each of them.

The difficulty is not building any one component. The difficulty is building 11+ interacting systems that work correctly together under production load, calibrated for safety-critical output, with learning loops that improve all components simultaneously from usage.

The safety calibration loop depends on having a safety detection system to calibrate. The safety detection system depends on a per-tenant sensitivity model. The per-tenant sensitivity model depends on interaction history. The interaction history depends on users trusting the safety warnings enough to dismiss or acknowledge them. That trust depends on the safety warnings being accurate. The accuracy depends on the calibration loop.

Circular dependencies of this kind don't have clean "build it and it will work" development paths. They close only through iterative production operation: each loop closure reveals new edge cases, the edge cases require system-wide adjustments, and the adjustments require revalidating the interactions between loops.

Accumulated Knowledge

The platform currently holds:

- 3,000+ documents in the acquisition pipeline (1,300+ indexed and searchable) across 127 manufacturers
- 187 equipment models with normalized metadata
- 120,000+ knowledge relationships
- 30+ autonomously generated research documents
- Glossary terms and equipment entity aliases extracted from every document
- Per-tenant safety calibration data
- Citation engagement histories across all queries
- Peer validation histories and equipment expertise badge progression
- Rudy Cards library with cross-tenant anonymized sharing data
- Stripe billing and conversion funnel data across consumer and B2B tiers
- A permanent, searchable corpus of real technician conversations — questions, answers, equipment context, and outcomes — accumulating from the first day of production usage

None of this was scraped from a public source. The calibration data, engagement histories, implicit feedback signals, and authority scores are artifacts of platform operation — they exist because the platform runs, and they grow because it keeps running. Every query adds engagement data. Every correction updates authority scores. Every safety interaction refines calibration. Every self-healing cycle adds research documents.

Trust Network

The 5-tier authority system represents accumulated trust that took verified contributions to build. Master-tier contributors have demonstrated domain expertise through approved corrections over time. That demonstrated authority is the basis for weighting their future contributions differently from a new account.

The value of the authority network — the signal that separates an expert correction from a guess — only develops through operation, and bootstrapping it artificially would defeat its purpose.

Trade-Specific Calibration

Generic configuration produces a functional system. Field-tested calibration produces a useful one.

The 27-cluster synonym expansion for BM25 queries was built from observed query patterns — terms that field technicians use interchangeably but that don't share lemmas. The 80+ term domain dictionary reflects abbreviations that appear in technician queries and must be expanded for accurate retrieval. The per-tenant safety calibration reflects actual dismissal patterns on actual equipment. The query reformulation strategies for each gap type were refined through production iteration.

This calibration is not documented anywhere. It exists in the system configuration, in the learned parameters of the feedback loops, and in the data accumulated from production operation. It cannot be reconstructed from first principles without operating the system at scale.

Quality Infrastructure

Most AI products ship features and hope they work. Rudy measures quality across dimensions most AI products don't measure at all — and the measurement infrastructure itself compounds: every evaluation cycle sharpens the next.

The lifecycle simulation framework runs 20 technician personas (from "apprentice, first week on job" to "30-year master tech, skeptical of AI") through compressed 60-day lifecycles. Each persona sends realistic queries, receives answers, and is evaluated daily on a retention meta-score. The dual-track judge measures both *technical correctness* and *relational quality* — catching the failure mode where a system gives correct answers that nobody wants to use because the interaction feels condescending or robotic.

Personalization delta testing runs paired accounts — one fresh, one with accumulated history — and samples on days 1, 15, 30, 45, and 60 to verify that returning users receive measurably better experiences than new users. If the personalization system is not producing a detectable delta, the compounding thesis is broken. This test catches that failure before users do.

The testing infrastructure (275+ automated tests across 3 tiers, safety testing framework (20+ scenarios with LLM judge and quality gates), production smoke suites, tRPC monolith-vs-Cloud-Run parity checks) ensures that quality cannot regress as new features ship. This removes the usual trade-off between shipping fast and maintaining quality.

Time

Compounding effects mean the platform's value grows with time, not just with effort. Each additional year of operation adds:

- 12 more months of citation engagement data in Rudy's ranking system
- 12 more months of authority scores accumulating across the contributor base
- 12 more months of safety calibration per tenant
- 12 more months of implicit feedback signals informing persona calibration
- 12 more months of self-healing research documents filling knowledge gaps
- 12 more months of cross-trade learning if Rudy has expanded to additional verticals

The gap is not linear. A knowledge base that has been operating for two years is not twice as valuable as one that has been operating for one year — it is materially more valuable, because the second year's learning builds on the first year's foundation, the community's authority scores are more differentiated, and the calibration has had more cycles to converge.

In compounding systems, time is an input: each month of operation builds on every month before it.

Part V: The Vision

The compounding engine described in Part III points toward two near-term directions that are not speculative — they are extensions of mechanisms already operating in production.

Personal Adaptation

The platform already learns about individual technicians through multiple mechanisms: which query patterns indicate expertise, which equipment this user works on, which implicit signals indicate frustration or satisfaction. Several personal adaptation features are deployed; the remainder are natural extensions.

Response calibration (deployed) — Per-user `detailLevel` preferences (brief/standard/detailed) persist across sessions and are wired into all synthesis prompts. Persona calibration infers expertise from query phrasing and adjusts response depth accordingly. A master technician asking about a complex fault receives a differential diagnosis tree; a junior technician asking the same question receives a step-by-step procedure with safety callouts at each step.

Equipment fleet memory (deployed) — Persistent equipment fleet profiles are live (PR #233, merged 2026-06-01). The `user_equipment_history` table tracks equipment with `status` and `pinned` columns. Fleet accumulates from usage and syncs the top-5 items to `profileData.memory.equipment`. A quick-pick bar on the chat interface and a dedicated My Equipment page at `/profile/equipment` let technicians manage their fleet. Fleet context is injected into synthesis prompts as `[TECHNICIAN CONTEXT]` for equipment disambiguation — replacing the previous 24-hour auto-load mechanism. When a service bulletin is ingested for equipment that appears in a technician's fleet history, the system can surface it without waiting for a query.

Proactive surfacing — The retrieval loop today is reactive: a technician asks, the system answers. With persistent equipment context and continuous knowledge ingestion, the system can alert technicians to relevant new information — manufacturer TSBs for equipment they service regularly, emerging failure patterns detected across the community, regulatory changes affecting their work.

Career Development Engine

The authority system was designed for content quality control. It also happens to be a competency record.

A technician who has been using Rudy for two years has a history of approved corrections, query patterns across equipment types, and demonstrated expertise across domains. That history is a competency profile that neither the technician nor their employer could easily construct otherwise.

Skill gap identification — If a technician regularly queries about procedures they should be performing independently, that is a signal about a training gap. If their query patterns on a specific equipment type are shifting from "how to" to "why does" questions, that is evidence of growing competency. The implicit feedback system already detects behavioral signals; the extension is interpreting those signals as competency indicators.

Growth trajectory — The authority tier system already tracks reputation. Making that trajectory visible to the technician creates a career development dimension that no other tool in their workflow provides.

Employer visibility (opt-in) — With technician consent, competency profiles can be shared with employers for workforce planning and training prioritization. The recruiting pipeline application — identifying technicians with demonstrated expertise in specific equipment types — is a direct extension of the authority system's existing data.

Trade Expansion

The 11-vertical expansion sequence is ordered by knowledge domain overlap — each trade inherits a substantial foundation from prior verticals, so each launch is faster and cheaper than the last:

#	Trade	Overlap	Why This Order
1	Generator / Power Systems	BASE	Beachhead — Cummins, Cat, MTU, Kohler, Generac
2	Diesel Truck Mechanic	~60%	Same engines (Cummins ISX, Cat C15, Detroit DD15), fuel, cooling, DPF/SCR
3	Marine Diesel	~55%	Same engine blocks with marine ratings. Tiny niche (~30-50K), dominate fast
4	Heavy Equipment	~50%	Cat + JD engines known. Hydraulics is the big new domain
5	Forklifts	~55%	Inherits hydraulics from #4, diesel from base. Small niche, few OEMs
6	Pump Technicians	~50%	Inherits hydraulics + industrial electrical. Bridges into plant operations
7	Crane & Rigging	~45%	Inherits hydraulics from #4. Highly specialized, high willingness to pay
8	Refrigeration / Cold Chain	~25%	Seeds the refrigeration domain before full HVAC. Commercial focus
9	Industrial Electrical	~40%	Electrical foundation from generators. Big market, enter after playbook proven
10	Elevator & Escalator	~30%	Builds on industrial electrical. Heavily regulated = sticky
11	HVAC & Refrigeration	~20%	Largest market (~400K). Refrigeration domain from #8. Pipeline battle-tested

Each vertical is not a separate product — it is a tenant configuration on the same platform, seeded with domain-specific content and safety calibration. The retrieval infrastructure, learning loops, quality evaluation, authority system, and multi-tenant architecture transfer without modification.

The compounding math: vertical #2 needs ~40% new content (60% reuse from generators). By vertical #5-6, the platform is mostly assembling from existing knowledge domains. By vertical #9-11, the document collection pipeline, self-healing knowledge loop, and user contributions are doing most of the work. The marginal cost of each new vertical declines as the platform matures.

Part VI: Future Ancillaries

The capabilities described in this section are natural extensions of the existing architecture, not separate product lines. They are presented as such because they require either user volume (to activate remaining learning loops) or hardware integration (to ingest physical-world data) that the current platform does not yet have.

Visual Diagnostics

Photo upload with Gemini multimodal analysis is already deployed in production. The extension is making this bidirectional: not just technicians sending photos to get answers, but an accumulating annotated image library that becomes a searchable visual reference.

Every photograph of a fault display, corroded connection, or failure mode — tagged with equipment, fault condition, and resolution outcome — builds a visual knowledge base that cannot be constructed from manufacturer documentation. Manufacturers document what components look like new. The visual knowledge base documents what they look like at every stage of failure.

Video-based procedure verification extends this further: a technician recording a maintenance procedure can have each step verified against the SOP, with missed steps and safety violations flagged in real time. Correct technique captured on video becomes training material directly from the field.

Acoustic Signature Analysis

Experienced technicians diagnose by ear. A bearing that is about to seize sounds different before it fails. Cavitation in a pump has a characteristic high-frequency signature. Misalignment creates specific harmonic patterns at rotation frequency and multiples. This diagnostic knowledge exists in experienced technicians' ears and nowhere else.

The approach is empirical: establish per-asset acoustic baselines during normal operation, detect deviations using spectral analysis, and validate failure-mode mappings through outcome tracking when deviations are subsequently confirmed as specific failure modes. Community validation means that each confirmed acoustic-to-failure mapping improves detection for every user on similar equipment.

IoT and Sensor Integration

Industrial equipment is already generating enormous volumes of operational data. SCADA systems, building management systems, genset controllers, PLCs, and IoT sensor networks produce continuous streams that go largely unanalyzed. MQTT, OPC-UA, Modbus TCP/RTU, BACnet, and REST-based APIs cover the major industrial communication protocols.

The value is context fusion: a temperature reading is not just a number — it is a reading from a specific sensor on a specific component of a specific unit with a specific maintenance history and OEM-defined operating limits. Correlating that reading with the maintenance history, the community's pattern database, and the equipment's known failure modes transforms an isolated data point into an actionable signal.

This direction is no longer purely speculative. Rudy's first hardware-integration endpoints are already in production (see *Platform API & Embedded Integrations* in Part II). A generator-monitoring product (Pulse) calls Rudy at device-commissioning time to resolve equipment profiles and diagnose fault codes against the knowledge base. These endpoints accept make/model identifiers and fault codes — not raw sensor telemetry yet — but they establish the stable API contract and authentication pattern through which richer sensor streams can connect as the platform matures.

Predictive Intelligence

The convergence of sensor telemetry, maintenance history, and community failure patterns creates the conditions for shifting from reactive support to predictive alerting. Individual sensor readings don't predict failures. Combinations of weak signals — temperature trending up 2°F per week, vibration shift on a specific bearing, query patterns from technicians who recently worked on similar units — collectively indicate an impending failure that no single signal would surface.

Condition-based maintenance scheduling is the direct economic case: replace time-based maintenance intervals with data-driven determinations of when maintenance is actually needed. The economic asymmetry is sharp — unnecessary maintenance is expensive; missed maintenance is catastrophic. Data closes that gap.

Frontier capabilities (longer-term): edge-deployed AI for sites without connectivity; natural language equipment control with human-in-the-loop confirmation and complete audit logging.

Part VII: Domain Expansion

The architecture's domain-agnosticism is not a marketing claim — it is a consequence of how the platform was built. The retrieval pipeline does not assume knowledge about any specific trade. The safety framework is a configurable pattern-matching system that accepts any hazard taxonomy. The authority system is a points-based tier mechanism with no domain-specific logic. The learning loops track signals that exist in any field service context: query satisfaction, citation engagement, correction authority, safety calibration.

Expanding to a new domain requires: domain knowledge seeding (equipment hierarchies, terminology glossaries, initial document corpus), safety pattern configuration for the domain's specific hazard categories, and authority hierarchy calibration appropriate to the trade's expertise structure. The retrieval infrastructure, feedback systems, quality evaluation, and admin tooling transfer without modification.

Domain	Knowledge Crisis	Multimodal Value	Predictive Value
HVAC	High tribal knowledge, fragmented manuals, same retiring-workforce dynamic	Acoustic: compressor health. Visual: refrigerant leaks, coil condition	Refrigerant pressure trends, energy consumption anomalies
Medical Equipment	Very high regulatory density, moderate tribal knowledge	Visual: error displays, wear indicators, calibration drift indicators	Sensor drift detection, calibration degradation prediction
Manufacturing	High tribal knowledge, many OEMs with proprietary diagnostics	Acoustic: motor and pump health. Visual: wear patterns, alignment	Vibration trends, production quality correlation with equipment state
Oil and Gas	High tribal knowledge, high regulatory, serious safety stakes	Acoustic: valve and pump cavitation. Visual: corrosion, joint integrity	Pressure and flow trends, wellhead performance degradation
Aviation MRO	Very high regulatory density, very high tribal knowledge	Visual: structural inspection, fastener condition. Acoustic: engine health	Flight data trends, component lifecycle analysis
Marine	Very high tribal knowledge, harsh environment failure modes	Acoustic: engine and hull. Visual: corrosion, bearing wear	Engine performance curves, hull fouling rate prediction

The key message for each vertical is the same as for power generation: the knowledge that keeps equipment running safely is fragmented, at risk of retirement, and impossible to capture with documentation alone. The architecture that solves this in power generation solves it structurally in every other trade. The cross-trade compounding from Part III means each additional vertical benefits from everything learned in all prior verticals.

About the Author

Nick Haschka is the founder of AskRudy and CEO of OnPoint Generators, Inc., a field service organization specializing in emergency backup power systems. With over two decades of operational experience in power

generation service, Nick has seen firsthand what the knowledge crisis costs: experienced technicians taking decades of diagnostic intuition into retirement, juniors learning through expensive callbacks, and customers bearing the cost of fragmented expertise.

Rudy began as an internal tool to make OnPoint's own technicians more effective. It became clear that the architecture being built was not solving a company-specific problem but an industry-wide one — and that the multi-tenant platform that emerged could serve any field service organization facing the same structural challenge. The same platform that helps an OnPoint technician in the field is equally available to any competitor willing to build their technicians' knowledge base on it, because the network effects that compound from a larger user base benefit everyone using the system.

The expansion thesis — from power generation to every trade where complex equipment requires expert diagnosis — follows directly from the architecture. The platform doesn't know it's about generators. It knows about knowledge retrieval, safety-critical information, expert authority, and continuous learning. Those properties transfer.

References

1. Lewis, P., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *NeurIPS 2020*.

2. NFPA 110: Standard for Emergency and Standby Power Systems. National Fire Protection Association.

3. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *EMNLP 2019*.

4. Es, S., et al. (2023). RAGAS: Automated Evaluation of Retrieval Augmented Generation. *arXiv preprint*.

5. Craswell, N., et al. (2020). Overview of the TREC 2019 Deep Learning Track. *TREC 2019*.

6. Nogueira, R., & Cho, K. (2019). Passage Re-ranking with BERT. *arXiv preprint*.

7. Mobley, R. K. (2002). An Introduction to Predictive Maintenance. *Butterworth-Heinemann*.

8. IEA World Energy Employment Report. International Energy Agency.

9. Goldman Sachs Global Investment Research. Power Generation Workforce Outlook.

10. U.S. Department of Labor, Bureau of Labor Statistics. Power Plant Operators, Distributors, and Dispatchers — Occupational Outlook Handbook.

AskRudy — OnPoint Generators, Inc. July 2026