RegenData

Mechanism-Anchored Constraint Governance for Physical Infrastructure AI

Technical Paper · Version 2

This paper is written for utilities, regulators, insurers, AI platform companies, and governance teams evaluating whether and how to constrain general-purpose AI models in physical infrastructure contexts.

1. Abstract

RegenData V2 presents a mechanism-anchored constraint governance architecture for AI assistants operating in physical infrastructure domains. The system externalizes admissibility, what AI is and is not permitted to say or recommend into a machine-readable domain canon, then enforces that canon at runtime through a deterministic execution boundary and a tamper-evident evidence layer.

The domain canon is RegenData's core asset. It encodes the physical mechanisms, regulatory limits, escalation requirements, and failure mode patterns of a specific infrastructure domain as machine-checkable, evidence-linked rules. The execution boundary is the hard gate that enforces the canon deterministically on every model output before it reaches an operator or user. The evidence layer produces contemporaneous, tamper-evident records of every boundary decision. The constraint story that regulators, insurers, and operators require.

This paper focuses on onsite wastewater treatment (OWTS) as the first validated domain. The same architecture extends to other water and wastewater domains, centralized treatment plants, drinking water networks, utility copilots by building additional domain canons on the same execution boundary and evidence infrastructure.

RegenData's claim is not that it makes AI models more intelligent. It is that a domain-specific, evidence-linked constraint layer external to the model, deterministic, and mechanism-anchored, is the missing component in every current AI deployment in physical infrastructure.

2. The Problem: AI in Physical Infrastructure

General-purpose AI assistants are being deployed into physical infrastructure at speed. Water and wastewater systems, energy networks, treatment plants, and utility operations are adopting AI copilots that generate recommendations, advise operators, and increasingly influence decisions with real physical consequences.

These models are capable. They produce fluent, contextually relevant responses. They draw on large bodies of domain knowledge. But they do not inherently distinguish between legally admissible, physically safe, and unsafe-but-plausible recommendations. A model trained on water engineering literature can generate a convincing dosing recommendation that exceeds the treatment capacity of a specific plant, ignores an irreversible failure threshold, or violates a permit condition specific to that facility.

In OWTS specifically the first domain in which RegenData has validated its architecture, systems are buried, heterogeneous, and often poorly documented. Homeowners ask AI assistants for help when something smells wrong, backs up, or seems off, rarely providing soil conditions, age, permit status, or inspection history. A seemingly reasonable AI response can encourage a homeowner to bypass a permit, defer a critical inspection, or attempt DIY repairs that damage the system or contaminate groundwater.

The core problem is the same across every physical infrastructure domain: AI systems can generate outputs that are linguistically plausible but physically inadmissible. Existing safety approaches reduce but do not eliminate this risk, and none of them produce the deterministic, evidence-linked constraint record that regulated infrastructure governance requires.

> *The operational question that existing approaches cannot answer: Is this specific output admissible under the physical and regulatory limits of this domain at this moment?*

3. Why Existing Approaches Are Insufficient

Existing AI safety approaches address important problems. They do not produce the constraint story that physical infrastructure governance requires.

3.1 The retrieval story

Retrieval-augmented generation with contextual grounding checks verifies that model outputs stay faithful to retrieved documents. This is a retrieval story: the system did not hallucinate beyond what it read. It cannot verify that what it read was correct, current, or applicable to the physical system's actual operating conditions.

In real infrastructure deployments, the RAG corpus is typically assembled by software engineers from whatever documents exist, old standards, vendor sheets, internal SOPs, regulatory documents of mixed vintage. Nobody canonizes that corpus before ingest. Nobody resolves contradictions between a 2019 standard and a 2024 revision. Nobody marks which guidelines apply only in certain jurisdictions. The grounding check faithfully enforces whatever the corpus contains including its errors, gaps, and outdated thresholds.

Documents also describe what to do. They rarely encode why the physical or biological mechanism behind the rule. A manual says dose at X rate. It does not encode: at higher rates the soil's aerobic zone collapses and treatment fails. The mechanism is not in the text. RAG retrieves the procedure. It cannot enforce the physical constraint behind it.

3.2 The probability story

Prompt engineering and system prompts shift the probability distribution of model outputs toward more compliant behaviour. Probability is not enforcement. A model soft-prompted with safety instructions remains capable of producing inadmissible outputs, particularly in edge cases, under adversarial inputs, or when instructions conflict. Prompt-based approaches produce no auditable record of which specific constraint governed a specific decision.

Fine-tuning adjusts model weights to make certain output patterns more or less likely. Fine-tuned behaviour degrades when the base model is updated or replaced. Fine-tuning cannot encode hard physical thresholds as deterministic constraints and produces no decision-level evidence chain.

3.3 The content story

Content filters and output classifiers pattern-match against known harmful content categories, toxicity, personally identifiable information, dangerous instructions by general standards. These filters have no awareness of domain-specific physical validity. A recommendation that exceeds the hydraulic loading threshold for a specific soil

classification passes every content filter. Content safety and physical safety are different problems solved at different layers.

3.4 The institutional governance story

Institutional governance frameworks including the EU AI Act, ISO 42001, NIST AI Risk Management Framework, ARAF, and the governance modules that AI copilot vendors are beginning to embed in utility products answer questions about accountability and process. Who authorized the system to operate. Who is responsible when something goes wrong. How decisions are documented. Whether the organization has met its regulatory obligations.

These frameworks are necessary. They do not answer the prior technical question: was this specific output physically and regulatorily admissible for this domain at this moment? That question requires domain-specific constraint architecture, not institutional governance. One does not substitute for the other; they operate at different layers of the stack.

3.5 What none of these produce

Each existing approach produces what can be described as a retrieval story, a probability story, a content story, or an institutional governance story. None produces a constraint story: a deterministic, mechanism-anchored record of whether a specific output was admissible under the physical and regulatory limits of a specific domain at a specific moment.

> *RAG plus institutional governance is not a complete solution for high-risk physical infrastructure. It is a retrieval story plus an accountability story. The constraint story is still missing.*

3.6 RegenData in the governance landscape

As institutional governance frameworks proliferate and copilot vendors embed governance modules in utility products, RegenData's position in the stack becomes clearer. RegenData is not a governance framework. It is the domain-specific constraint layer that makes the outputs being governed physically valid in the first place.

A governance framework without a domain canon governs process without physical validity guarantees. A domain canon without a governance framework enforces physical constraints without institutional accountability. Together they address both requirements. RegenData is designed to operate alongside any institutional governance framework. Its execution

boundary is exposable as a permit/block API, allowing any governance platform to route physical infrastructure AI traffic through domain-specific canons rather than ad-hoc filters.

As the governance landscape matures, RegenData's irreducible core, the domain canon and its deterministic enforcement at the execution boundary, becomes more valuable, not less. Every governance framework that emerges creates demand for the constraint layer that makes the outputs being governed physically admissible.

4. Core Architecture

RegenData's architecture has three inseparable components and one internal process.

The three components are the irreducible core of any RegenData deployment:

- The domain canon - the mechanism-anchored, evidence-linked, versioned encoding of what AI is and is not permitted to say or recommend in a specific physical infrastructure domain.
- The execution boundary - the hard gate that enforces the canon deterministically against every model output before it reaches an operator or user.
- The evidence layer - the contemporaneous, tamper-evident record of every boundary decision, inseparable from the execution boundary.

The internal process is canon stewardship - RegenData's methodology for maintaining and improving canons over time based on block patterns and expert review. This process operates at RegenData, not at the client deployment. Clients receive improved canon versions as outputs of this process.

The model itself is unchanged. Any general-purpose AI model can be used, GPT-family models, Claude, or others. The model is a probabilistic proposal engine. The canon and execution boundary decide what proposals are admissible. The model can propose anything. The boundary decides what reaches the user.

4.1 The domain canon

The domain canon is the single source of truth for admissibility in a specific physical infrastructure domain. It is not a document, a prompt, or a retrieval corpus. It is a structured set of machine-checkable, evidence-linked rules that encode what is physically true and legally permitted in that domain.

The canon is built from first principles. In OWTS, those first principles are: septic systems are living systems, more like wetlands than machinery and pipes; and septic systems are flow-through systems; what goes in moves through the system and ultimately reaches the environment. These truths shape every node, rule, and governed response in the canon. They cannot be derived from a document corpus; they must be encoded by someone who understands the domain at the mechanism level.

The canon consists of five interlocking components:

• Domain nodes - semantic units representing mechanisms, components, and contexts (e.g., "conventional septic tank," "pressure distribution," "high water table," "suspected hydraulic overload").

- Rules - machine-checkable constraints tied to nodes, specifying admissible and inadmissible statements, required caveats, and escalation triggers.
- Failure categories - normalized labels for common unsafe patterns (e.g., "unsafe DIY repair," "permit bypass," "insufficient information," "suspected public health risk").
- Governed responses - pre-authored message templates mapped to failure categories and rules, written in user-appropriate language and approved by domain experts.
- Evidence links - references into design manuals, regulations, field data, and expert judgments that justify each rule.

Together these components define the semantic surface on which admissibility decisions are made. The canon becomes a formal specification of what "safe to say" means in a physical infrastructure domain.

4.2 The provenance store

Each rule in the canon is linked to the evidence that justifies it through a provenance store; a structured repository maintaining bidirectional relationships between rules and their authoritative sources. Sources include regulatory documents, design standards, field studies, and expert judgments.

The provenance store serves two functions at evaluation time. First, during evidence map lookup, it supplies the authoritative context behind rules being evaluated, enabling the constraint engine to apply rules with explicit reference to their justification rather than as bare assertions. Second, it ensures every block decision in the evidence layer carries a traceable citation; the specific regulation, standard, or field observation that makes the blocked output inadmissible.

When a regulator, insurer, or operator asks not just what was blocked but why that action was inadmissible/admissible, the provenance store provides the answer. It also supports canon maintenance: when a regulation changes or new field data challenges an existing

threshold, the provenance store identifies which rules are affected and what evidence needs to be updated.

The provenance store is versioned alongside the canon. Every canon release includes a snapshot of the provenance relationships in force at that version, ensuring the justification for any historical decision can be reconstructed regardless of subsequent canon updates.

4.3 The execution boundary

The execution boundary is the hard gate. It is a separate component, external to the model, that intercepts every model output and evaluates it against canon-derived rules before the output reaches the user or downstream system. It asks one narrow question:

> *Is what the model proposed to say admissible under the current domain canon?*

At runtime the surface is intentionally simple: permit or block. Blocks are always paired with governed responses rather than silent failure. The boundary evaluates multi-part responses at the segment level each claim or recommendation is checked against relevant canon nodes independently. Permitted segments pass through intact. Blocked segments are replaced inline. The user or operator receives a single coherent answer combining permitted content and governed responses wherever the draft violated the canon.

Every block produces two simultaneous outputs. The first is the user-facing governed response, a standardized, defensible message replacing the inadmissible content. The second is a governance output, a block notification carrying the constraint details: what was recommended, which canon rule fired, which failure category was triggered, and the evidence chain that justifies the constraint.

The routing of the governance output is deployment-specific.

In homeowner deployments such as the OWTS reference implementation, the governance output routes exclusively to the evidence layer. The homeowner receives the governed response. The technical constraint details are not surfaced to the user but remain fully available for audit, canon stewardship review, and regulatory inspection.

In utility operator deployments, the governance output surfaces to the operator in addition to routing to the evidence layer. Operators need to know what was recommended, why it was blocked, and which physical or regulatory constraint was violated. Not only in the next audit cycle but at the moment of the decision. For high-risk block categories, recommendations approaching irreversible failure thresholds, permit-sensitive decisions, dosing levels near critical operating limits, or outputs with potential public health consequences, the block notification triggers an immediate escalation pathway. The operator receives the constraint details in real time. The evidence layer records everything simultaneously.

This distinction between deployment contexts is a configuration of the execution boundary, not an architectural change. The boundary always produces both outputs. What differs is where the governance output goes and how urgently it is surfaced.

The boundary does not redefine policy, improvise domain judgment, or invent authority. Its role is intentionally narrow. Narrow systems are easier to audit, test, and govern. Because it is small and deterministic rather than large and probabilistic, the boundary can be unit-tested, regression-tested, and formally verified where warranted, the same way any safety-critical control system is treated.

The boundary is model-agnostic. It operates on outputs, not on model weights or prompts. It works identically whether the upstream model is GPT-4, Claude, Gemini, or any future general-purpose model. This model-agnosticism is what makes the canon licensable as infrastructure rather than as a model-specific patch.

4.4 The evidence layer

The evidence layer is inseparable from the execution boundary. Every permit or block decision writes a contemporaneous, tamper-evident record simultaneously with the routing decision, not after, not asynchronously, but at the moment of the decision.

Each record includes:

- User input
- Model draft output (or hashed reference)
- Applicable rules and nodes
- Decision outcome (permit / block)
- Reason code(s)
- Canon version in force
- Timestamp
- Deployment and tenant identifiers where relevant

This is the constraint story. Not what the model read. Not how likely the output was. Not whether it contained toxic content. But whether this specific output was admissible under the physical and regulatory limits of this domain at this moment, with the specific rule that fired, the evidence that justified that rule, and the canon version that was in force.

These records enable reconstruction of specific decisions, audits of rule application consistency, statistical analysis of block patterns over time, and the contemporaneous tamper-evident documentation that regulators, insurers, and courts will expect when AI-assisted decisions in regulated infrastructure are scrutinized.

Authority over what the canon is lives outside the runtime, in a domain governance process involving subject-matter experts, regulators, operators, and insurers. The boundary merely enforces

the canon currently in force. Every decision is tagged with the canon version that applied, making it possible to explain changes over time as the canon evolves.

5. Canon Stewardship

Canon stewardship is RegenData's internal process for maintaining and improving domain canons over time. It is not a client-facing layer. Clients receive updated canon versions as outputs of this process. The process operates at RegenData.

Block events are the primary input to canon stewardship. Because the model is constrained by canon context before generation, blocks in well-tuned deployments are rare. Each block is therefore a high-value governance signal, an indication that the canon, a governed response, or the interaction between model behaviour and canon rules may need attention.

Canon stewardship operates through two review functions:

5.1 Stabilization

Stabilization review is triggered when a specific reason code recurs above a defined threshold across block events in the evidence layer. A domain-qualified reviewer examines the blocked outputs, the rule that fired, and the governed response that replaced the unsafe content. The reviewer determines whether the block pattern reflects a canon gap requiring rule refinement, a governed response gap requiring template improvement, or an expected and correct enforcement pattern requiring no change.

Proposed changes are documented with the evidence that triggered them and submitted for canon version approval.

5.2 Coherence

Coherence review is conducted on a scheduled cadence typically monthly or quarterly in early deployments, examining the distribution of permit and block decisions over time. Coherence asks whether the overall enforcement pattern remains consistent with the canon's original safety intent, even when individual decisions are locally defensible. Significant shifts in block rates, reason code distributions, or node activation patterns trigger structured review regardless of whether any individual block exceeded the local Stabilization threshold.

5.3 Canon version governance

The learning loop produces proposed canon changes, not autonomous canon updates. Every proposed change moves through a defined human governance process before becoming part of the canon version enforced at runtime.

Canon version approval requires sign-off from the designated Canon Owner and Safety Lead. Changes are tagged with the evidence that justified them, the reviewer who proposed them, and the approver who authorized them. This governance trail is stored alongside the canon version history, ensuring the justification for every canon change is permanently recoverable.

6. Architecture Stack

The RegenData stack for any domain deployment has four layers.

6.1 Domain canon layer

The domain canon is stored in a machine-readable format and versioned under human governance as described in Section 5. It is the only element shared across all deployments within a domain. The canon is RegenData's core deliverable and primary asset.

6.2 Model layer

Any general-purpose AI model can be used. The model is unchanged by RegenData. It receives the user's query and generates a draft output. It is a probabilistic proposal engine. It is not the governance mechanism.

In deployments where the model already receives rich domain context, SCADA data feeds, digital twin state, RAG corpora, or fine-tuned domain knowledge, that context stack is unaffected by RegenData. The execution boundary evaluates whatever the model produces, regardless of how the model was grounded. RegenData complements existing context infrastructure; it does not replace or compete with it.

6.3 Execution boundary

The execution boundary is middleware that intercepts every model output and evaluates it against canon-derived executable rules, producing a deterministic permit or block decision. It operates exclusively on model output, not on the user's question. The question is never blocked; only inadmissible draft content is.

6.4 Evidence layer

The evidence layer is an append-only, tamper-evident log of decisions as described in Section 4.4. It is co-located with the execution boundary and writes simultaneously with every boundary decision. Block events are surfaced for canon stewardship review.

6.5 Current implementation status

RegenData V2 describes a target architecture in which the execution boundary is implemented as deployed middleware, a separate microservice intercepting every model output at runtime before it reaches the user.

In the current implementation, enforcement functions are delivered through structured constraint prompts injected into the model's system prompt at inference time. Canon rules, escalation patterns, and governed response templates are assembled into prompt slices that bias the model toward admissible outputs before generation. The boundary then evaluates the draft output against these injected constraints, producing permit or block decisions.

This approach is sufficient for the reference implementation described in Section 8 and for the validation results in Section 8.1. It does not yet provide the cryptographically verifiable, atomic evidence records that production deployment in regulated infrastructure requires.

The transition from constraint-prompt enforcement to production middleware is the primary engineering objective of RegenData's co-development pilot phase. The pilot implementation will deploy the execution boundary as a dedicated microservice, implement atomic record-writing to the evidence layer, and produce decision records satisfying the contemporaneous, tamper-evident standard described in Section 4.4.

This distinction is stated explicitly so that organizations evaluating RegenData for regulated infrastructure deployment understand the current stage and the pathway to production readiness.

7. OWTS as First Domain

Onsite wastewater treatment is the first domain in which the RegenData canon has been fully specified, built, and tested against real-world question patterns.

OWTS was chosen because it concentrates the properties that make physical infrastructure AI governance difficult: systems are heterogeneous and poorly documented; ownership is decentralized; the users asking questions, homeowners, typically have no technical training; the information environment is polluted with manufacturer marketing, unqualified forum advice, and AI assistants that sound authoritative but have no physical constraint layer; and the failure states are irreversible at the individual, community, and watershed level.

If constraint governance works in OWTS, the hardest context in the water infrastructure family, the architecture generalizes. OWTS is the stress test that validates the pattern.

The OWTS canon encodes:

- Physical mechanisms - how soil-based treatment actually works, what degrades biological function, what causes irreversible failure.
- Regulatory thresholds - permit requirements, design standards, inspection obligations, and jurisdictional variations.
- Failure mode taxonomy - the categories of unsafe AI output specific to this domain (unsafe DIY repair, permit bypass, insufficient information, out-of-scope, suspected public health risk).
- Escalation patterns - when AI must stop advising and direct the user to a licensed professional or regulatory authority.
- Governed responses - pre-authored, expert-approved templates for every blocked failure category.

7.1 Reference implementation

The primary reference implementation is an OWTS assistant answering homeowner and small-system operator questions about septic systems and related decentralized infrastructure. The assistant operates as follows:

1. User submits a free-form question - often ambiguous, emotional, and incomplete.
2. The model generates a draft answer.
3. The execution boundary evaluates the draft against the OWTS canon, permitting or blocking segments.
4. The user receives a governed answer - permitted content combined with governed responses wherever the draft violated the canon.
5. A decision record is written to the evidence layer.

This deployment is intentionally narrow: one domain, one class of users, a constrained set of integrations. Its purpose is to validate the canon, boundary, and evidence layer in a real question distribution before extending to other water and wastewater contexts.

7.2 Validation results

Initial validation was conducted under the current constraint-prompt implementation. The validation used a structured test set of OWTS-relevant homeowner questions spanning the canon's primary node categories — system maintenance, additive use, failure recognition, DIY repair, permit-sensitive modifications, and escalation scenarios.

Each question was submitted to the same general-purpose model in two conditions: unconstrained, and with canon constraint prompts applied. Conditions were evaluated independently to preserve scoring integrity.

Outputs were evaluated against a defined safety rubric using a blind scoring methodology. A general-purpose AI model was used as the independent scorer — evaluating each response without knowledge of whether it was produced under constrained or unconstrained conditions. This approach was chosen to eliminate the confirmation bias inherent in having the canon's author or a canon-familiar expert evaluate outcomes they could identify by condition. Each response was scored for physical admissibility, regulatory compliance, and appropriate escalation behaviour against the rubric criteria.

> *Canon-constrained outputs demonstrated a 76% reduction in unsafe recommendations compared to unconstrained outputs across the controlled test set.*

Unsafe recommendations included physically inadmissible advice, permit bypass encouragement, unsafe DIY guidance, and failure to escalate in high-risk scenarios.

This result validates the canon's node structure, rule coverage, and governed response design as effective constraints on general-purpose model behaviour in the OWTS domain. It confirms the core thesis: that a mechanism-anchored domain canon, applied as a constraint

layer over model outputs, meaningfully governs model behaviour in physical infrastructure contexts.

The blind GPAI scoring methodology strengthens the validity of this result. The scorer had no awareness of which condition produced which output, no familiarity with the canon's intent, and no stake in the outcome. The 76% reduction therefore reflects a genuine signal in the outputs rather than a scoring artefact.

This is not a longitudinal field study or a production-grade safety certification. The test set, while structured and representative of real OWTS question distributions, is controlled rather than drawn from live production traffic. Production middleware deployment, intended for co-development pilots, will enable more rigorous instance-level validation with tamper-evident decision records and real operator question distributions.

8. Beyond OWTS: Toward a Physical Infrastructure Canon Portfolio

The OWTS canon demonstrates the architecture in a decentralized, homeowner-facing domain where systems are heterogeneous, documentation is sparse, and the consequences of unsafe AI advice are irreversible at the individual and community level.

The same problem exists in every physical infrastructure domain where AI is being deployed. The admissibility question is identical. The architecture is the same. Only the canon changes.

8.1 Centralized water and wastewater utility operations

Centralized water and wastewater utility operations present a structurally analogous challenge at a different scale. Where OWTS deals with individual homeowner decisions, utility operations deal with operator decisions affecting entire communities, treatment plant dosing, pump station management, network pressure regulation, permit compliance, and emergency response protocols.

A utility AI copilot advising on chlorine dosing rates operates in a domain governed by maximum residual disinfectant levels, contact time requirements, disinfection byproduct formation limits, and permit conditions specific to the treatment plant and its source water. A recommendation that is plausible in general terms may be inadmissible for a specific plant operating near a permit threshold, drawing from a source with elevated organic precursors, or responding to an unusual demand event.

No RAG corpus encodes those plant-specific admissibility conditions deterministically. No fine-tuned model enforces them as hard constraints. No institutional governance framework answers whether the specific dosing recommendation was physically valid before it reached the operator.

A centralized water and wastewater canon would encode the mechanism-anchored constraints governing these decisions, treatment chemistry boundaries, hydraulic operating envelopes, permit thresholds, escalation requirements, and failure mode recognition patterns. The execution boundary and evidence layer require no architectural changes. The same hard gate that enforces the OWTS canon enforces the centralized water canon. The domain knowledge lives in the canon. The enforcement and evidence infrastructure is shared.

The centralized water and wastewater canon described here is illustrative. The OWTS canon is the only domain canon currently validated. Extension to centralized water and wastewater operations is the objective of RegenData's co-development pilot program.

8.2 Shared scaffold and domain customization

Extending RegenData across multiple deployments within a domain does not require rebuilding the canon from scratch for each client. The canon architecture supports a shared scaffold model in which approximately 80% of the canon, core physical mechanisms, regulatory thresholds, failure categories, escalation patterns, and governed responses applicable across the domain, is maintained centrally and shared across all deployments.

The remaining 20% is customized per deployment to reflect client-specific operating conditions: local regulatory variations, facility-specific capacity parameters, organizational authority structures, proprietary operational data, and jurisdiction-specific permit requirements. This customization layer is hosted within the client's own infrastructure, ensuring sensitive operational data never leaves the client's environment.

This scaffold model has three practical consequences. First, it reduces the canon development burden for each new deployment, partners contribute domain expertise to validate and extend the shared scaffold rather than building from first principles. Second, it creates a network effect in which improvements to the shared scaffold benefit all deployments simultaneously. Third, it makes the commercial structure of co-development pilots coherent: the pilot funds the development of a domain scaffold that becomes a shared asset, with per-deployment customization as the ongoing engagement model.

8.3 Canon portfolio model

In the canon portfolio model, RegenData maintains one execution boundary architecture and one evidence layer infrastructure. Multiple domain canons are developed with the institutions that own those systems. utilities, regulators, research consortia, engineering firms. Each canon is domain-specific. The enforcement and evidence infrastructure is shared.

Extending RegenData beyond OWTS requires:

- New domain canons - for centralized water treatment, drinking water networks, reuse systems, stormwater, energy systems, and related domains, developed with domain experts and validated against real operational question distributions.
- Co-development partnerships - domain canons cannot be built without deep domain knowledge. Each new domain requires a co-development partner who brings operational expertise, regulatory fluency, and validation capability.
- Shared boundary and evidence infrastructure - the same execution boundary and evidence layer are reused across domains with domain-specific rules and governed responses.
- API integration - the execution boundary can be exposed as a permit/block API, allowing any AI governance platform or copilot vendor to route physical infrastructure AI traffic through domain-specific canons rather than ad-hoc filters.

RegenData's core asset across all domains is the canon and its methodology: turning physical mechanisms, regulatory limits, and escalation practices into machine-checkable, evidence-linked rules that a deterministic execution boundary can enforce at runtime.

9. Conclusion

Physical infrastructure AI is not a content problem. It is a constraint problem.

Generic models can already generate convincing, contextually relevant advice for water and wastewater operators, homeowners, and engineers. Existing safety approaches, RAG, prompt engineering, fine-tuning, content filters, and institutional governance frameworks reduce but do not eliminate the risk of physically inadmissible outputs. None of them produce the constraint story that regulated infrastructure governance requires: a deterministic, mechanism-anchored record of whether a specific output was admissible under the physical and regulatory limits of a specific domain at a specific moment.

RegenData V2 proposes a different relationship between AI systems and domain authority:

- The model remains a probabilistic proposal engine.
- The domain canon defines admissibility for a specific physical infrastructure domain.
- The execution boundary enforces the canon at runtime - external to the model, model-agnostic, deterministic.
- The evidence layer records what fired and when - contemporaneously, tamper-evidently, with the specific rule and provenance that justified every decision.
- Canon stewardship maintains and improves the canon over time through block-driven review and human governance - at RegenData, not at the client.

In OWTS, this pattern turns a complex mix of physical mechanisms, regulations, and escalation practices into something a governance-minded organization can inspect, audit, and improve. The same pattern extends to centralized water, wastewater treatment, drinking water networks, and every other physical infrastructure domain where AI is being deployed without the constraint layer it requires.

> *The goal is not merely safer AI outputs. It is to make AI governance in physical infrastructure traceable, insurable, and accountable, by putting a mechanism-anchored domain canon, not the model, in charge of what is allowed to reach the operator.*

RegenData's irreducible core is the domain canon and its enforcement at the execution boundary. As governance frameworks proliferate and context infrastructure in utility deployments matures, this remains the non-substitutable layer, the only component in the physical infrastructure AI stack that produces a deterministic, mechanism-anchored constraint story.