FRANK GOMEZ

Principal Data Systems Engineer

Location: Remote / Las Vegas

Contact: f.source@outlook.com

Control-First Architecture · Decision Correctness · System-Level Ownership

(Distributed Systems, Observability, Automation, Control)

Designs, operates, and governs production systems prioritizing correctness, auditability, and failure containment.

POSITIONING SUMMARY

Principal data systems operator specializing in the design, operation, and governance of production systems where correctness, auditability, and failure containment are non-negotiable.

Work centers on complex, distributed environments that must remain legible under stress, inspectable after failure, and governable as scope increases, with decisions designed to be traceable, bounded, and defensible.

OPERATING POSTURE

I operate with a control-first, evidence-driven posture when designing and running systems. Decisions favor bounded execution, explicit constraints, and rollback-safe paths over unchecked autonomy or speed.

Systems are built to surface meaningful signals early, contain blast radius when failures occur, and remain legible and debuggable as complexity and pressure increase. Intervention is designed to be safe, auditable, and intentional, not reactive or ad hoc.

OPERATING PROFILE

- Operates at principal / CTO altitude while retaining the ability to enter detail selectively, without collapsing into local optimization

- Owns ambiguous, cross-system problem spaces end-to-end (definition → constraints → execution → rollback)
- Functions as connective tissue across engineering, data, operations, and leadership, aligning incentives, signals, and decision horizons
- Defaults to long-horizon system correctness over short-term feature velocity, even under delivery pressure
- Maintains high signal-to-noise communication: precise framing, bounded options, explicit tradeoffs
- Escalates deliberately: intervenes when systemic risk appears, stays hands-off when systems are behaving within bounds
- Capable of rapid context-switching between strategic framing and tactical inspection without losing global state

CONTROL ASSUMPTIONS & CONSTRAINTS

The systems I build assume that failure, ambiguity, and change are normal operating conditions, not edge cases. Control exists to make those conditions survivable, inspectable, and correctable.

Decision Accountability

- Every material decision has a trace: who approved it, what inputs were used, and what constraints applied at the time
- Systems are designed so decisions can be revisited without reconstructing history from memory

Observability as a Baseline Requirement

- Systems are assumed untrustworthy without direct, continuous visibility
- Metrics, logs, and state must exist before scale, optimization, or automation

Rollback as a First-Class Capability

- All deployments, configuration changes, and automated actions assume reversibility
- No system is considered “safe” unless rollback is faster than forward repair

Bounded Authority

- Automation operates within explicit limits
- Humans remain the final authority on actions with irreversible impact

Failure Containment

- Faults are expected and isolated
- Blast radius is constrained by design, not by reaction

Change Discipline

- Changes are gated, observable, and auditable
- Speed is never prioritized over recoverability

Ownership Clarity

- Every system, pipeline, and control surface has a clear owner
- Responsibility is explicit, not inferred

OPERATING PRINCIPLES & PROOF

Principle 1: Observability is a control primitive (no silent failure).

Proof: Systems treated missing telemetry as a failure class; operators diagnosed system state through correlated signals already present, not emergency instrumentation or interpretive debate.

Principle 2: Decision integrity requires evidence, not narrative (decision traceability by default).

Proof: Delivered decision surfaces grounded in measurable health, risk, and capacity signals; implemented traceable decision lineage enabling post-incident reconstruction of who decided what, when, and why.

Principle 3: Execution must be bounded: constraint-first design with explicit authority.

Proof: Architected systems around known limits, invariants, and failure envelopes before features; implemented policy-bound action surfaces where high-risk operations were mechanically constrained, not trust-based.

Principle 4: Auditability is a feature: systems explain themselves.

Proof: Built traceable action → state → outcome chains ("who / what / when / why") enabling forensic reconstruction without dependence on institutional memory or narrative justification.

Principle 5: Rollback is not optional: reversibility is the default safety guarantee.

Proof: Shipped production changes with deterministic rollback paths, versioned state transitions, and rehearsed recovery procedures used during real incidents.

Principle 6: Gating enforces correctness: validation before and after change prevents drift.

Proof: Applied verification gates (pre-change checks, runtime validation, post-change reconciliation) to prevent silent regressions and narrative-driven releases.

Principle 7: Automation must reduce risk, not amplify it (governed automation).

Proof: Implemented gated, auditable automation with approval paths, kill-switches, and rollback-safe execution; prevented cascading failures from unbounded automation.

Principle 8: Complexity must be localized and blast radius bounded.

Proof: Designed architectures that isolated failure domains and constrained impact; systems degraded explicitly and predictably rather than failing globally or silently.

EXECUTIVE CONTROL LAYERS

Complex systems under my ownership are structured into explicit control layers.

Each layer **bounds complexity, enforces correctness, and preserves decision integrity at scale.**

This is a **governance model**, not an implementation inventory.

1. System Architecture & Platform Design

Owns and governs system boundaries, dependency direction, coupling decisions, failure domains, and evolution paths to preserve long-term maintainability and keep complexity governable over time.

2. Data, Streaming & Decision Infrastructure

Designs and governs how signals are produced, propagated, validated, and transformed into explainable decisions, so correctness is verifiable over time under uncertainty and load.

3. Operational Control Surfaces & Dashboards

Builds and governs control surfaces that let operators and executives observe, intervene, and make time-sensitive decisions over live systems safely, without relying on tribal knowledge.

4. Observability, Reliability & Incident Control

Owns system behavior under stress by governing detection, diagnosis, and recovery to prevent silent failure, enable deterministic response, and drive institutional learning.

5. Automation, Orchestration & Delivery Systems

Replaces manual process with repeatable, gated, auditable automation that controls change execution, validation, and rollback so speed never compromises safety.

6. Cloud, Runtime & Infrastructure Substrate

Governs execution environments and underlying resources so behavior is predictable, isolation is enforced, scaling is controlled, and failures are contained beneath application logic.

7. AI / LLM / Agentic Systems & Intelligence Layer

Embeds intelligence into systems with bounded authority and autonomy, preserving auditability, evaluability, safety, and human accountability.

Layer Detail

Each layer is independently governable and collectively composable.

Expanded architectures, substrates, responsibilities, and proof signals for all seven layers are provided in the companion document:

→ ***Executive Control Layers: Architecture & Substrate Breakdown***

CORE CAPABILITIES

The following competencies describe how execution occurs inside the control domains above, independent of title, company, or toolchain. These are repeatable operating behaviors, not role claims.

Control-First System Design

- Architect systems with explicit control boundaries, not implicit coupling
- Design for governability first, feature velocity second
- Encode ownership, escalation paths, and rollback authority into the system itself
- Balance modularity with operability (avoid "clean architecture" that is un-debuggable)

Decision-Grade Signal Engineering

- Build pipelines that convert raw events into decision-grade signals
- Distinguish storage, analytics, and decision systems, and keep them separate
- Design for latency awareness, freshness guarantees, and signal confidence
- Treat absence-of-signal as a first-class failure mode

Operability & Human Control

- Create control surfaces where humans can see, diagnose, and intervene safely
- Design dashboards as decision instruments, not status walls
- Ensure operators can trace why something happened, not just what it did
- Bind telemetry, logs, and state into coherent operational narratives

Governed Automation & Intelligence

- Delegate execution to machines without delegating accountability
- Encode preconditions, invariants, and rollback paths into automated workflows
- Build automation that fails safely, noisily, and reversibly
- Treat automation as a managed actor, not a fire-and-forget script
- Embed reasoning systems where they augment control, not replace it
- Ground intelligence outputs in verifiable context and bounded action spaces
- Design AI-enabled systems with evaluation, auditability, and fallback paths
- Treat AI as an operator-extender, not an authority substitute

PROOF ANCHORS

A recurring set of structural characteristics that appear wherever I am responsible for system design, governance, or execution, independent of domain, team, or tooling.

These are inspection handles. If the claims above are accurate, the traits below should be observable.

Decision Traceability

- Major decisions leave durable artifacts: logs, state transitions, approvals, rollbacks
- It is possible to answer why a system behaved a certain way, not just what it did
- Decision paths remain reconstructible without relying on memory or tribal knowledge

Operability Under Stress

- Systems remain diagnosable during partial failure
- Operators can intervene without unsafe shortcuts or hidden state
- Recovery paths are exercised and validated, not theoretical

Auditability & Accountability

- Ownership boundaries are explicit and encoded into workflows
- Actions are attributable to identifiable actors (human or automated)
- Critical paths avoid black-box execution without inspection capability

Signal Quality Discipline

- Metrics are selected for decision usefulness, not vanity
- Missing, delayed, or contradictory signals are handled explicitly
- Alerting surfaces actionable anomalies rather than noise

Automation with Reversibility

- Automated actions are gated by explicit preconditions and invariants
- Rollback paths exist and are faster than forward repair
- Automation failures fail visibly and safely

Governance by Design

- Guardrails are embedded structurally, not enforced socially
- Permissioning, approval, and escalation are part of system architecture
- Systems default to safe states when assumptions break

Cross-System Consistency

- Control patterns recur across unrelated systems
- Recovery logic, decision framing, and control surfaces are recognizable
- New systems inherit constraints rather than reinventing them

These anchors provide inspection handles. Case studies demonstrate what occurred; these patterns describe what must consistently be true across systems if the operating claims above are accurate.

CASE STUDY INDEX

The following case studies provide concrete examples of how the operating posture, control assumptions, and executive control layers outlined above were applied under real system pressure.

Each case focuses on:

- System-level decision making under constraint
- Tradeoffs involving reliability, observability, and control
- Failure modes, recovery paths, and governance considerations
- Outcomes that could be falsified, audited, or reproduced

Read these when evaluating:

- Decision-making under uncertainty
- Failure modes, containment, and recovery
- Constraints, tradeoffs, and outcomes
- Judgment, resilience, and execution under ambiguity

These case studies are recommended for:

- Senior technical reviewers
- Architecture or platform interviews
- Roles where system correctness and operational discipline are critical

They are intentionally separated from this document to preserve clarity and signal-to-noise ratio.

→ ***Operating Principles & Proof***

CROSS-CASE SYNTHESIS

Across the case studies, a consistent set of system-level patterns emerges, independent of domain or implementation detail.

These patterns are not project-specific lessons; they represent repeatable invariants observed under real operational pressure:

- Decision quality degrades predictably when signal quality is incomplete, delayed, or unauditable.
- Systems without explicit control surfaces force operators into unsafe, ad-hoc intervention.

- Failures rarely originate from single components; they emerge from gaps between observability, ownership, and actionability.
- The absence of telemetry or audit trails is itself a failure mode, often more dangerous than visible errors.
- Human-in-the-loop governance is essential at escalation boundaries; fully automated systems without checkpoints amplify risk.
- Rollback capability and decision traceability determine whether incidents are survivable or terminal.

These invariants inform the operating posture, control assumptions, and executive control layers described in this document.

SELECTED SYSTEM THEMES

- **Control surfaces over raw systems**: prioritizing governed interfaces, visibility, and safe intervention over opaque complexity
- **Signal quality over feature velocity**: optimizing for observability, correctness, and decision clarity rather than output volume
- **Absence of signal as signal**: treating silence, gaps, and negative space as first-class diagnostic inputs
- **Human-in-the-loop governance**: deliberate points of review, override, and accountability where automation meets risk
- **Explicit boundaries and failure modes**: systems designed with known limits, blast radii, and degradation paths
- **Decision traceability over intuition**: decisions that can be reconstructed, audited, and corrected post-hoc
- **Rollback-first design**: favoring reversible actions and controlled unwind over brittle “heroic” execution

ROLE FIT (TARGET ROLE ALIGNMENT)

This profile maps best to roles where system correctness, control, and long-term operability matter more than feature velocity alone.

Strong Fit For

- Principal / Staff Engineer roles with cross-system ownership
- Founding or scaling CTO roles where architecture, governance, and decision integrity are core responsibilities
- Senior technical leadership positions in:
    - Complex platforms
    - Data- and decision-heavy systems
    - High-reliability or regulated environments
- Organizations where:
    - Failure modes must be understood, not discovered accidentally
    - Operators require clear control surfaces
    - Technical decisions have long-lived consequences

Weaker Fit For

- Feature-only execution roles with no system ownership
- Environments optimizing solely for short-term output
- Teams without tolerance or appetite for explicit control, auditability, or constraint-driven design

This section exists to reduce hiring ambiguity, not to broaden appeal.

RELATED ARTIFACTS

This document is the **primary control artifact** and is designed to **stand on its own** as a control-oriented, principal-level resume.

The materials below provide **alternate inspection angles** where useful; they do **not** replace this document.

Concise Summary Resume

A compressed, executive-level overview of this operating profile.

Designed for rapid inspection and first-pass evaluation, it presents the control posture, executive domains, and technology surface in a tightly condensed format.

Best used when:

- A shorter submission is required
- A high-level capability snapshot is preferred
- Initial screening precedes deeper technical review

→ ***Summary Resume***

Structured Systems Resume

A structured resume-format presentation of system ownership, scope, and operating posture, designed for recruiter and hiring-manager review.

- Organized in conventional résumé sections for fast scanning
- Emphasizes responsibility, scope, and system-level impact
- Frames control architecture within recognizable role expectations

This artifact provides a familiar entry point for standard hiring workflows while preserving the integrity of the operating model.

→ ***Structured Systems Resume***

Historical Portfolio Resume Archive

Two previously published resume artifacts reflecting earlier leadership roles in design, marketing, and systems strategy.

- Structured visual formats with full portfolio integration
- Design-forward presentation and narrative positioning
- Cross-functional leadership across product, brand, and execution

These materials represent prior career phases and are provided for historical and structural context.

→ ***Resume Archive***

EXECUTIVE CONTEXT

This work reflects principal- and executive-level ownership of complex, high-leverage systems where correctness, resilience, and decision traceability matter more than feature velocity. The emphasis is not on isolated technical achievements, but on sustained system stewardship: designing environments where operators can reason clearly, intervene safely, and recover decisively under real operational pressure.

The role assumed across these systems is not that of an implementer alone, but of a systems owner responsible for aligning architecture, control surfaces, operational discipline, and risk boundaries with executive intent. This includes translating ambiguous business goals into enforceable system constraints, ensuring decisions remain auditable and reversible, and maintaining clarity across teams as systems scale in complexity and consequence.

This context is intended for readers evaluating senior technical leadership (Principal, Staff, Head of Systems, or CTO-level roles) where the primary value lies in judgment, system integrity, and the ability to govern complexity over time.

SCOPE

Open to senior, high-leverage roles where system correctness, operational discipline, and long-term maintainability matter more than short-term feature velocity.

Best suited for environments that value:

- Clear ownership and decision boundaries
- Measurable system behavior
- Explicit tradeoffs over implicit assumptions

This document is intended to stand on its own.

Additional materials are provided for depth where useful.