Panoramic View: System Cartography Field Manual

Author: Mike Hall (Staff Software Engineer) · **Canonical Route:** https://www.just3ws.com/panoramic-view/

System Cartography is the empirical discipline of discovering, mapping, and reconciling how a complex platform actually operates. Panoramic View reconciles a system as interested parties understand it with the domain model that encodes its business concepts and the runtime evidence of how it actually behaves. Follow any actor's journey from left to right; at every step, trace the interaction through the system and back with the same fidelity.

The unit of inquiry is the intersection of business process and system operation. As time and mandate permit, the map can extend beyond an interface path to upstream supply chain, external partners, operational controls, retention, and archival cold storage.

Origin & Flagship Context

Mike Hall conceived Panoramic View circa 2021-2022, drawing on years of production diagnosis, legacy-system ownership, observability, incremental modernization, and organizational knowledge recovery. The earliest timestamped exported evidence is from September 2023.

From desktop browser to mainframe and back again. The flagship proving ground was a high-volume personal lending platform. Mike mapped the entire loan acquisition and origination pipeline end-to-end from initial visit to booked loan, conducting longitudinal analysis across the process sequence and latitudinal analysis from desktop browser down through Rails services and MuleSoft integrations to IBM mainframe backends and back again at each step. The Affiliate API was mapped with the same depth and process.

Three Accounts of One System

- **Interested-party understanding:** The process, outcome, rules, and responsibilities customers, operators, Product, leadership, and other stakeholders believe the system supports.
- **Implemented domain model:** The business concepts, relationships, constraints, decisions, and state transitions encoded in software and data.
- **Observed runtime behavior:** What source, state changes, traces, logs, integrations, and external effects show the system actually does.

Panoramic View maps the correspondences and gaps among all three. The goal is a complete-enough account for the defined decision scope, with contradictions, confidence, provenance, and unknowns still visible.

The Four Dimensions

- **The journey moves across (longitudinal) (Horizontal axis):** The actor enters stage left, moves through a business process sequence, pauses or branches, and eventually exits stage right with an outcome. Longitudinal analysis maps this horizontal progression end-to-end.
- **Each interaction moves down and back (latitudinal) (Vertical axis):** A choice descends from the interface through application, service, integration, and data layers. The response rises through those layers and becomes the next state presented to the actor. Latitudinal analysis traces this full-stack depth at each step.
- **Evidence and meaning travel together (Context axis):** Runtime signals, source history, recordings, state changes, and human knowledge are correlated with interested-party understanding, the implemented domain model, actual system operation, ownership, confidence, contradictions, and explicit unknowns.
- **Time and impact measure knowledge, not just latency (Temporal & impact axis):** The time dimension extends beyond network round-trips into context loading costs, institutional knowledge attrition, and the operational radius of impact. Business decisions are captured as tracked assets, anchoring infrastructure investments to the sustained value of the underlying business processes they support.

The 5-Step Discovery Loop

1. **Define the premise:** Name the actor, intended outcome, starting state, and question the investigation must answer.
2. **Traverse the next interaction:** Follow the request and response through every observable technical and business layer.
3. **Cross the boundary:** Bring in the person who understands the next ownership, domain, or system boundary.
4. **Capture evidence:** Record behavior, state changes, transformations, decisions, absences, provenance, and confidence.
5. **Synthesize and repeat:** Update the model, mark contradictions and gaps, then continue until the required path is navigable.

One Method, Four Proof Terrains

- **OneMain Financial (Enterprise Legacy Lending):** Mapped 7 acquisition funnels end-to-end from browser through Rails and MuleSoft to IBM z/OS mainframe; eliminated a 4% silent e-sign cookie overflow via zero-downtime DynamoDB migration; engineered a 5-phase PII purge across 30+ tables.

- **media-pipeline (Autonomous Media Ops):** 11-checkpoint synthetic tracer, OMG BPMN 2.0 subprocess orchestration, explicit failure offramps, and honest calibration snapshots.
- **Phalanx Duel (Deterministic Game Engine):** Living architecture with 12 wiki diagrams, a Groma C4 explorer, and automated freshness gates synchronized to production TypeScript and Go code.
- **zdots & just3ws.com (Personal Control Plane & Archive):** Observable Zsh control plane (founded May 2017) serving as the root platform for Panoramic View Labs (PVL), paired with adots, vdots, and a 185-interview oral history datalake.

Field Vocabulary

- **The system already has a rhythm.** (*Operational cadence, feedback control, and phase alignment*): Observe the repeated movement before changing it. Look for drift, breaks, and changes in tempo.
- **Follow the actor's journey through the system.** (*Longitudinal analysis, service blueprinting, and value-stream mapping*): Connect the actor's goal and visible experience to the work, decisions, and systems behind it across the process sequence.
- **Each choice creates a wave down and back.** (*Latitudinal analysis, distributed tracing, and request-response flow*): Trace a request down through interfaces, application code, services, integrations, data, and the response that returns back.
- **Compare what people believe with what the system does.** (*Model reconciliation and socio-technical sensemaking*): Place stakeholder understanding, the implemented domain model, and runtime evidence beside one another.
- **Bring in the person who understands the next boundary.** (*Boundary spanning and cross-functional collaboration*): Treat ownership and context as part of the system map, not as private knowledge outside it.
- **Give different groups a map they can use together.** (*Boundary object*): Create a shared artifact that Product, Engineering, Operations, and leadership can each read from their own perspective.
- **Observe, adjust, and observe again.** (*Closed-loop control and double-loop learning*): Change the action, then question the assumptions and goals when the result does not match the intended outcome.
- **Learn what normal looks like before extracting anything.** (*Behavioral baseline and service characterization*): Measure the current pattern before modernization so the later change can be compared with evidence.
- **The system can run too hot or too cold.** (*Control band, setpoint, deadband, and SLO boundaries*): Define a useful operating range with upper and lower signals instead of treating only failure as important.
- **Keep uncertainty visible.** (*Provenance, confidence tracking, and explicit unknowns*): Record where a claim came from, how strong it is, and what remains unresolved.

- **Communication paths become architecture.** (*Conway's Law and socio-technical congruence*): Compare the dependencies in the software with the communication and ownership structure around it.
- **Hotpath mapping is one view of the larger map.** (*Critical-path analysis and journey prioritization*): Use frequency, business importance, or risk to choose where to begin without mistaking the first slice for the whole system.
- **Business decisions are tracked assets that set the radius of impact.** (*Decision asset tracking, radius of impact, and critical context bounds*): Capture business decisions as concrete, tracked system assets mapped to code paths, data structures, and runtime infrastructure. This bounds the radius of impact and preserves the critical context required for safe change.
- **Infrastructure value mirrors the business decisions it sustains.** (*Value-derived infrastructure, holistic investment, and socio-technical stewardship*): Map key decision points to the systems they run on to evaluate infrastructure value directly against business capability value. Sustained investment supports both the computing foundation and the long-term human capacity required to operate it.
- **The time dimension measures context load and knowledge rot, not just API latency.** (*Cognitive latency, context loading costs, and knowledge attrition prevention*): Evaluate the time dimension of transactions across the entire lifecycle, including the hours spent recovering lost architectural context, the acceleration of knowledge rot, and the true cost of unmapped dependencies.
- **A map that aligns human teams grounds AI agents too.** (*Agentic grounding, socio-technical cartography, and hallucination containment*): Use the same empirical system map that bridges Product and Engineering to bound AI agent context. Anchoring agents in verified decision points eliminates context thrashing and prevents speculative code generation.