Excellent â€” below is the full text for: /root/harmonix/docs/HMX_MissionControl_DesignDoc_v1.2.md (âœ… Commit-ready, Markdown format, with embedded Mermaid visuals, glossary, and visual stack references). â¸» ğŸ›°ï¸ Harmonix Operator Console â€” Mission Control Telemetry Architecture Version: 1.2 (Enhanced Visual Stack Edition) Date: 2025-10-27 Authors: Data Blocks | Harmonix Division Status: Draft â€” Open for Review â¸» 1. Executive Summary The Harmonix Mission Control Telemetry Architecture transforms the Operator Console into an intelligent visualization and diagnostic system for Akash-based infrastructure. It delivers complete visibility from bare metal through Kubernetes and Akash runtime layers, enabling predictive insights, replay-based troubleshooting, and rapid root-cause analysis. Goal: reduce operational downtime by 40% and improve decision accuracy through data-driven visualization, modular telemetry, and replay analysis. â¸» 2. Purpose and Vision Mission Control evolves the Operator Console into a multi-layer observability and replay platform, capturing system data across the full lifecycle of Akash deployments. Quantitative Objectives: â€¢ Support replay of 7+ days of node activity with < 5 % performance overhead. â€¢ Visualize live and archived metrics with < 1 s telemetry latency. â€¢ Integrate seamlessly into Harmonix self-healing workflows. â¸» 3. System Overview Layer Purpose Source Example Metrics Hardware Physical server telemetry IPMI, lm-sensors CPU temp, fan speed OS Resource utilization psutil, top CPU %, memory, uptime Kubernetes Cluster state kube-api, metrics-server Pod health, restarts Akash Runtime Blockchain activity provider-services, RPC Bid count, lease state Harmonix Self-healing & orchestration HMX daemons Probe results, recovery logs â¸» 4. Data Collection Pipeline Stage Description Error Handling Collection Scrape metrics from subsystems 3 retries w/ exponential backoff Normalization Convert to JSON schema Drop invalid or partial frames Storage Write to /metrics/ and /data/archives Rotate every 24 h Distribution Serve via NGINX + JSON endpoints Validate MIME and syntax Visualization Render in Operator Console panels Fallback to cached replay data â¸» 5. Data Flow & Telemetry Bus sequenceDiagram participant HW as Hardware participant OS as OS Collector participant K8s as Kubernetes participant HMX as Harmonix Runtime participant UI as Operator Console HW->>OS: Sensors, CPU, Temp OS->>K8s: Aggregated Metrics K8s->>HMX: Pod + Node State HMX->>UI: Telemetry JSON (WebSocket) UI->>HMX: Control & Acknowledgement â¸» 6. Visualization Framework Panels are modular and responsive, powered by lightweight JavaScript modules and dynamically updated via WebSockets or replayed JSON. Key Modules: â€¢ runtime_badge.js: Node status, uptime, resource load â€¢ preflight_badge.js: Validation status before deploy â€¢ rules_panel.js: Security and compliance rule viewer â€¢ api_integrity_card.js: Endpoint health overview â€¢ endpoint_shim.js: Replay fallback logic Accessibility: ARIA-compliant labels, high-contrast color scheme, and keyboard navigation are standard. â¸» 7. Replay & Flight Recorder Feature Description Snapshot Frequency 10 min (configurable) Archive Retention 7 days (~10 MB / hour) Replay Format JSON manifest + timestamped bundles Security SHA256 file integrity + TLS delivery Usage Root-cause analysis, performance tuning, compliance verification â¸» 8. Software Stack Layer Technology Rationale OS Ubuntu 22.04 LTS Stability and LTS support Kubernetes k3s v1.34.1 Lightweight cluster management

Telemetry psutil, jq, curl Portable data collection Web Server NGINX 1.18 + gzip Reliable static + JSON serving Visualization Vanilla JS + Mermaid + Chart.js Zero dependencies, GitHub renderable Docs MkDocs + Material Theme Modern navigation and visuals â¸» 9. Deployment Strategy gantt title Harmonix Mission Control Roadmap dateFormat YYYY-MM-DD section Data Collection Hardware & OS Metrics :done, 2025-10-15, 5d Kubernetes Telemetry :active, 2025-10-20, 7d section Visualization Mission Control Dashboard : 2025-10-27, 10d Replay & Flight Recorder : 2025-11-05, 7d section Integration Akash Runtime + Security : 2025-11-12, 5d Final QA & Docs : 2025-11-18, 4d Milestones: Phase Deliverable Owner Date 1 Golden Node Telemetry Capture Harmonix Core 2025-10-30 2 UI Visualization Panels Operator Console Team 2025-11-05 3 Replay Flight Recorder Data Blocks Labs 2025-11-12 4 Akash Runtime Integration Provider Ops 2025-11-20 â¸» 10. Risks, Assumptions, and Dependencies Category Description Mitigation Risk Akash RPC latency > 5 s Implement local caching Risk Provider JSON corruption Real-time JSON linting Assumption Ubuntu 22.04 availability Mirror backup Dependency GitHub + Cloudflare uptime Offline replay fallback â¸» 11. Testing and Validation Test Criteria Method Unit JSON schema validity jq â€“e . Integration End-to-end telemetry replay Simulated failure replay Performance < 1 s latency, < 5 % CPU overhead stress-ng, top User Acceptance Dashboard consistency Manual UI regression test â¸» 12. Goals for Support Input & Next Steps â€¢ Open for peer review via GitHub PR. â€¢ Solicit input on telemetry schema extensibility and visual layout. â€¢ Next milestone: integrate with Harmonix Runtime auto-heal logs and publish /docs site on Cloudflare Pages. â¸» Appendix A â€” Glossary Term Definition Akash Provider A decentralized compute node offering resources to tenants via Akash Network. Tendermint Byzantine Fault Tolerant consensus engine underpinning Akash blockchain. Helm Chart Kubernetes packaging format defining services and deployments. Telemetry Bus Harmonix JSON transport channel between daemons and UI. Harmonix Flight Recorder Archival replay system capturing runtime metrics for post-event analysis. â¸» Appendix B â€” Visual Standards and Tools Category Tool Type Integration Diagrams Mermaid.js Open-source Inline in Markdown Wireframes Figma / Penpot Free plan / Open source Export SVG â†’ /docs/assets Charts Plotly.js / Chart.js Open-source Used in dashboards Docs Rendering MkDocs Material Open-source / docs/ site PDF Export Pandoc + WeasyPrint Free pandoc *.md -o *.pdf --css=docs.css Automation GitHub Actions CI/CD Auto-build & artifact export Diagrams Draw.io (diagrams.net) Free Editable .drawio source 3D Visuals Blender Open-source Optional renders for hero visuals Visual Style: Dark theme Â· Accent #4DD6FF Â· Font Inter/Roboto Â· Max width 1200 px Â· High contrast. â¸» Appendix C â€” JSON Schema Example { "version": "1.0", "timestamp": "2025-10-27T00:00:00Z", "cpu_pct": 0.64, "mem_pct": 3.06, "load1": 0.35, "uptime_sec": 425122, "status": "ok" } â¸» Appendix D â€” Changelog Version Date Summary 1.0 2025-10-26 Initial draft 1.1 2025-10-27 Added Executive Summary, Testing, Risks 1.2 2025-10-27 Enhanced Visual Stack Edition â€” Mermaid + MkDocs + Glossary â¸» âœ… End of Document â¸»