T6 — Petter Graff master synthesis

ALAI Holding AS — Chief Architect Master Synthesis

MC #10357 T6 | Petter Graff, Lead Architect

Date: 2026-04-30 | Status: FINAL

SECTION 1: EXECUTIVE SUMMARY (CEO read-first, 60 seconds)

What you have: A sophisticated single-machine AI orchestration system running on ANVIL (Mac Studio). The core — Claude Code as orchestrator, 40 sub-agents, 10,220 tracked tasks, 138 LaunchAgent daemons, Litestream backup, local Ollama fleet — is structurally sound. Better than most funded startups at similar stage. **The three risks that matter:**

1. **Legal exposure right now.** 84 signed corporate PDFs, NDAs, and insurance documents have NEVER been uploaded to cold storage. `legal-docs-azure-sync` has been failing silently since deployment. If you need to produce a document for Akershus funding review or any legal dispute, you are relying on local disk only.

2. **$129K/day burn with zero revenue.** 91% is Opus. The cost-tracker numbers need unit verification — if these are actual dollars (not micro-cents), you have 2-3 weeks of runway at this rate before any individual account limit triggers. A runaway Opus loop at midnight runs for 8 hours before anyone sees it.

3. **Azure VM is an unprotected single point of failure.** BookStack (your operational wiki), Vaultwarden (every system password), Documenso, Grafana, Planka — none have ANVIL-side backup. If Azure deletes that VM tonight, the knowledge base and all credentials are gone.

What to do tomorrow morning: 1. Fix `legal-docs-azure-sync.sh` — add `--overwrite` flag (15-minute fix) 2. Verify cost-tracker units against actual Anthropic billing dashboard 3. Enable Azure VM-level backup policy from Azure portal (30-minute click job) 4. Run: `node ~/system/tools/cost-tracker.js summary today` and pin the number

The architecture is salvageable. The process discipline needs surgery, not the infrastructure.

SECTION 2: CURRENT-STATE ARCHITECTURE DIAGRAM

```
============================================================
LAYER A — ANTHROPIC ORCHESTRATION SURFACE
============================================================
```

CEO Prompt | v [John — Claude Code, Sonnet-4.6] | +-- UserPromptSubmit hooks (6 hooks, cache-defeating): | boot-enforcer.sh | validation-state-injector.sh <-- INJECTS mc.js list EVERY prompt | alai-hooks auto-verify (Kotlin CLI, 14MB binary) | alem-instruction-checker.sh | feasibility-check-advisory.sh | incident-response-mode.sh | +-- /mehanik (pre-dispatch gate, slash command) | -> ~/.claude/agents/mehanik.md [model: sonnet — OVERTIRED; should be haiku] | -> writes /tmp/mehanik-cleared-<MC_ID> | -> pre-dispatch-gate.sh validates 13-field marker | +-- Task tool dispatch (PreToolUse hook stack): lock-john-dispatch-cap.sh [cap=3; TOO LOW — should be 8] john-max-depth-gate.sh [ZAKON #28, trip-wires 1+2: CORRECT] pre-dispatch-gate.sh pre-action-da-gate.sh | v Sub-agent (isolated context, ~/.claude/agents/<name>.md) | 40 .md files — 22 doing real work, 8 umbrella personas (ROUTING TRAP), 5 reviewer duplicates, 7 mapped-but-missing (BROKEN DISPATCH) | v /task-postflight --> Proveo verify --> mc.js done alai-hooks evidence-gate enforces postflight marker

== = LAYER B — ORCHESTRATION KERNEL (~/system/kernel/) == =

pi-orchestrator.js [5,251 LOC MONOLITH] Status: IDLE since 2026-04-28 22:24Z (48+ hours) Blocked by: 3 tasks stuck in ready_for_review (#10038, #10039, #10043) Owns: 1,619 tasks (1,324 paused + 295 blocked) — NO PICKUP MECHANISM | +-- 30s poll loop -> getNextTask() [non-atomic 3-statement claim — RACE CONDITION] +-- model selection (tier 1-5) -> Ollama (FORGE/ANVIL) or Claude API +-- quality gate (lint/test/semantic) +-- proof-of-work verification +-- HiveMind posting (fire-and-forget, NOT events.db) +-- cost tracking +-- YouTube RAG ingest [verified Ollama qwen3:8b, ZERO Anthropic cost]

cross-company-bus.js [6h cron] 85 matches today, 1 task created Enforces cross-company-routes.json (10 rules, no real-time hook)

chain-runner.js [YAML chains, lightly used] sprint-pipeline.js [DAG state in JSON column — cannot SQL-query] pipeline-controller.js [13-phase lifecycle]

DEAD COMPONENTS (marked): [DEAD] agent-orchestrator.js — archived 2026-03-21 (was never canonical) [DEAD] HelixSupport — merged into FlowForge; stale ref in cross-company-routes.json [DEAD] Proxima — merged into Lexicon; stale refs exist

==========
= COMPANY / AGENT ROUTING LAYER
==========
=

specialist-mapping.json — 28 agents mapped, 12 companies 17 ORPHAN agents (in ~/.claude/agents/ but NOT in mapping) 7 BROKEN entries (in mapping but .md file does NOT exist) Datavera: ZERO agents (ML/analytics domain UNROUTED)

Cross-company routing (bi-modal only): WRONG: tier-router.js — only "ollama" | "cc" | "human-queue" — no OpenAI, no Gemini OK: comms-responder.js — groq -> claude-api -> claude-cli -> ollama (only multi-vendor path) OK: rag-router.js — cache -> local-ollama -> local-enriched -> external (Claude fallback)

discover.js routing "<query>" — primary routing query tool Returns only indexed agents; 17 orphans are INVISIBLE to this tool

==========
= DATA LAYER
==========
=

mission-control.db [25.6 MB main + 23.3 MB WAL — needs checkpoint] 10,220 tasks | 14 concurrent writers | NO command authority boundary Non-atomic task claim (3 separate DML statements, no BEGIN TRANSACTION) Missing index: tasks(delegated_to) — hot query path unindexed 27 dangling FK references (PRAGMA foreign_keys NOT enforced) version column exists for optimistic locking — enforcement unverified

hivemind.db [60.1 MB main + 147 MB WAL — CRITICAL BLOAT] WAL = 245% of main file. Multi-second startup delay on crash restart.

flywheel.db [224.2 MB main + 264 MB WAL — HIGH BLOAT] WAL = 118% of main. Continuous Litestream replication of dirty WAL. flywheel.db.interactions has used_for_training column — distillation HALF-BUILT

events.db [14.6 MB + 4.5 MB WAL — acceptable] Schema: idempotency_key UNIQUE, correlation_id, causation_id — CORRECTLY DESIGNED Problem: PRIMARY orchestration path BYPASSES this store (posts to HiveMind instead)

ingest-queue.sqlite.corrupt [~/system/state/ — evidence of past corruption, no RCA]

Litestream replication: 50+ databases -> Azure Blob (alaibackups0ebb.blob.core.windows.net) P0 critical: 1s sync, 7d retention azure-db-backup.sh: PARTIAL FAILURE (intermittent timeout on 400M+ blobs) RESTORE DRILL: NEVER PERFORMED

==========
= DAEMON FLEET (138 LaunchAgents)
==========
=

47 running with PIDs 69 calendar-OK (scheduled, idle) 4 HARD ERROR (exit 256 every run): [FAIL] com.alai.azure-db-backup — LightRAG blob upload timeout (intermittent) [FAIL] com.alai.apply-knowledge — needs investigation [FAIL] com.john.auto-verify-regression — MC #10173 closed but still failing [FAIL] com.john.infra-drift-detector — git rename limit (3178 files), one-liner fix

CRITICAL LEGAL FAILURE: [FAIL] com.john.legal-docs-azure-sync — exit 512, BlobAlreadyExists (missing --overwrite) 84/84 legal documents NEVER uploaded. Has NEVER worked.

NOT LOADED: com.alai.lightrag-auto-heal — plist exists, not bootstrapped com.john.rdap-audit-quarterly

FALSE ALARM FLOOD: com.john.ops-watchdog — docs.basicconsulting.no (302=live), vault.basicconsulting.no (302=live) flagged as DOWN 10+ consecutive checks. Real outage (lumiscare.no) buried in noise. lumiscare.no: fail 2500/2 — GENUINE outage, invisible due to noise.

MC #10173 FRAUD: Marked done 2026-04-30T03:55Z. 4 of 11 original failures STILL FAILING at 19:50Z.

==========
= VENDOR COUPLING
==========
=

Anthropic: 9/10 lock-in (Claude Code IS the orchestrator runtime) $129K/day (243 req), $368K/week (1,429 req) — 100% claude-cli 91% of requests are Opus (176 req today at ~$730

avg) UNIT VERIFICATION REQUIRED against actual billing

Local Ollama: 6/10 dependency (FORGE 10.0.0.2:11434 + ANVIL localhost:11434) 7 models on FORGE, 8 on ANVIL, ~143GB allocated MLX servers: :11435 gemma-4-26b, :11436 qwen3-32b, :11437 qwen3-coder-30b Fine-tuned: alaiml-task-v1, alaiml-email-v1, alaiml-tender-v1

Google Gemini: 1/10 (gemini-reviewer.md only, free tier) Groq: 2/10 (comms-responder fallback only) OpenAI: 0 requests ever — 0% integrated, 8/10 missed opportunity

==========

SECTION 3: TOP 10 PROBLEMS (Ranked by L×S/E)

Scoring: L = Likelihood (1-5), S = Severity (1-5), E = Effort-to-fix (1-5 hours) Higher score = worse and cheaper to fix = fix first.

P0 — THIS WEEK (Fix or accept the consequence)

#1 — legal-docs-azure-sync: 84 legal documents never uploaded

Score: L=5, S=5, E=1 → **25.0 Teams:** T5 (primary), also violates T1 ZAKON ARCHIVE FIRST **Evidence:** `legal-docs-azure-sync.log` shows `ERROR: BlobAlreadyExists` on all 84 files, plus ISBD source directory does not exist. Exit 512 every run. Has never succeeded. **Risk:** If Akershus funding review, investor due diligence, or legal dispute requires document production — reliance on ANVIL local disk only. If ANVIL fails without verified Azure backup, documents are gone. **Fix:** Add `--overwrite` flag to `az storage blob upload-batch` in `/Users/makinja/system/daemons/legal-docs-azure-sync.sh`. Fix ISBD path. Verify next run. Estimated: 15 minutes.

#2 — Cost units unverified: $129K/day could be catastrophic or harmless

Score: L=3, S=5, E=1 → **15.0 Teams:** T4 (primary), T5 (flagged) **Evidence:** `cost-tracker.js summary today` reports $129,209.12 today, $368,509.48 this week. These numbers come from ALAI's internal logger (claude-cli backend). It is NOT verified against Anthropic's billing dashboard. **Risk:** If actual dollars: 2-3 weeks until account-level limits trigger. No revenue to offset. If micro-cents: the architecture is fine but the CEO is making decisions on phantom numbers. **Fix:** Login to console.anthropic.com, compare actual billing charges for this week to `cost-tracker.js` output. One manual verification, 10 minutes. Then add a monthly reconciliation task.

#3 — Azure VM: BookStack + Vaultwarden have NO backup

Score: L=2, S=5, E=2 → **5.0 Teams:** T5 (primary), T2 (cross-reference) **Evidence:** `litestream.yml` covers `~/system/databases/` on ANVIL, not `4.223.110.181`. `azure-db-backup.sh` backs up ANVIL Docker volumes, not the VM. `dr-sync.sh` syncs to FORGE (10.0.0.2), not Azure VM. **Risk:** Azure VM at `4.223.110.181` runs BookStack (operational wiki — 100% of your system documentation), Vaultwarden (every API key, every credential), Documenso, Grafana, Planka. VM deletion = total knowledge and credential loss. Vaultwarden contains the keys needed to rebuild everything. **Fix:** Enable Azure VM Backup policy from Azure portal (30-minute click job). OR add a nightly `mysqldump` + `sqlite3 .dump` pull from VM to ANVIL via SSH cron.

#4 — MC #10173 closed under false pretenses: 4 of 11 failures STILL FAILING

Score: L=5, S=3, E=2 → **7.5 Teams:** T5 (primary), T3 (process failure pattern) **Evidence:** Fleet snapshot at 19:50Z (16 hours after #10173 marked done) shows `calendar_err_256` on `auto-verify-regression`, `infra-drift-detector`, `apply-knowledge`. `legal-docs-azure-sync` still exit 512. **Risk:** This is primarily a process integrity problem. Hard Constraint #3 ("Builder cannot say done") was violated. The 4 remaining failures compound daily. infra-drift-detector has a one-liner fix (git config). **Fix:** Reopen MC #10173. Fix infra-drift-detector (add `git config diff.renameLimit 0`). Investigate apply-knowledge. Do NOT mark done until fleet snapshot confirms 0 errors on all 4.

#5 — WAL bloat: 411MB uncheckpointed across hivemind.db + flywheel.db

Score: L=4, S=3, E=1 → **12.0 Teams:** T2 (primary) **Evidence:** hivemind.db-wal = 147MB (245% of main), flywheel.db-wal = 264MB (118% of main). PRAGMA wal_checkpoint returned 0 pages checkpointed for mission-control.db. **Risk:** On unexpected restart, SQLite replays entire WAL before any operation. For hivemind.db, this means 147MB replay — multi-second to minute-level startup delay. Under full-disk condition, WAL growth can prevent writes to ALL SQLite databases sharing the volume. **Fix:** Add a 4-hour LaunchAgent running `PRAGMA wal_checkpoint(TRUNCATE)` on hivemind.db, flywheel.db, and mission-control.db. Create ~/system/tools/wal-checkpoint.js. Estimated: 2 hours.

P1 — THIS MONTH

#6 — ops-watchdog false alarms drowning real outages

Score: L=5, S=3, E=1 → **15.0 Teams:** T5 (primary) **Evidence:** `docs.basicconsulting.no` and `vault.basicconsulting.no` both return HTTP 302 (services live) but watchdog treats 302 as failure. Both show `fail 10/2`. Meanwhile `lumiscare.no` shows `fail 2500/2` — genuine 2500-check outage buried in noise. **Risk:** Watchdog loses credibility. Real production outages go undetected because every alert is assumed false positive. Lumiscare has been down for 2500 consecutive checks (2-minute interval = 83 hours of outage) with no escalation. **Fix:** Add `--location --max-redirs 1` to curl checks in ops-watchdog script. Accept 2xx/3xx as healthy. Investigate lumiscare.no DNS/CF issue separately. 1 hour.

#7 — 17 orphan agents + 7 broken mapping entries

Score: L=5, S=2, E=3 → **3.3 Teams:** T1 (primary), T3 (confirms) **Evidence:** 17 agents in `~/.claude/agents/` are invisible to `discover.js routing`. 7 mapping entries (`hadi-hariri`, `lee-robinson`, `james-bach`, `lisa-crispin`, `dorota-huizinga`, `maria-santos`, `resolver`) point to .md files that do not exist — any routing to these crashes at dispatch. **Risk:** John bypasses routing and dispatches manually. Programmatic routing (cross-company-bus, pi-orchestrator) cannot see 37% of agents. When pi-orchestrator resumes, it will route to broken entries and generate task failures. **Fix:** Add 17 orphans to specialist-mapping.json. Delete or stub-create 7 broken entries. Run `~/bin/agent-definitions-sync.sh`. Estimated: 3 hours.

#8 — Non-atomic task claim: latent double-claim race condition

Score: L=2, S=4, E=2 → **4.0 Teams:** T2 (primary) **Evidence:** The claim path in pi-orchestrator.js is 3 separate DML statements (lease_until update, status=in_progress, active_work insert) with no wrapping transaction. If process crashes between statements 1 and 2, state is inconsistent. With 14 concurrent writers to mission-control.db, the race window is real. **Risk:** With current pi-orchestrator idle this is latent not active. When it resumes at scale, double-claims can cause duplicate work, wasted Opus spend, and corrupted task state. **Fix:** Wrap the 3-statement claim in `db.transaction(() => { ... })()` using better-sqlite3's synchronous transaction API. Also add `db.pragma('foreign_keys = ON')` per connection. Add missing index `idx_task_scheduling_next_eligible`. Estimated: 1-2 days.

#9 — Dispatch cap 3/session: too restrictive for legitimate workflows

Score: L=5, S=2, E=1 → **10.0 Teams:** T3 (primary), T4 (confirms) **Evidence:** `lock-john-dispatch-cap.sh` blocks 4th Task tool invocation per session. One legitimate task = build + Proveo + Skillforge = 3 dispatches. A second task in same session is immediately blocked. Today's 5-team audit required `[CEO_APPROVED]` override. **Risk:** John is forced to use `[CEO_APPROVED]` as a workaround for normal workflows. This defeats the purpose of ZAKON #28 tracking — legitimate overrides obscure actual drift events. Anthropic's own SDK design assumes multi-sub-agent parallel dispatch. **Fix:** Edit `~/.claude/hooks/lock-john-dispatch-cap.sh` line 77: change `MAX_DISPATCHES=3` to `MAX_DISPATCHES=8`. Also add skillforge, proveo to bootstrap-exempt list. Single-line change. 10 minutes.

#10 — validation-state-injector defeats prompt prefix cache every session

Score: L=5, S=2, E=2 → **5.0 Teams:** T3 (primary), T4 (cost context) **Evidence:** `validation-state-injector.sh` runs `mc.js list --owner john --priority H --json` on every UserPromptSubmit (30s TTL cache). Dynamic output changes whenever task state changes, invalidating Anthropic's automatic 5-min ephemeral cache from that breakpoint forward. On a 4-hour session with frequent task state changes: ~50% cache miss on content that should be static. **Risk:** Direct cost amplifier. T3 estimates ~$2.27/4-hour session in preventable input token charges. At daily usage, this compounds. More importantly, it is architecturally inverted: ambient state does not belong in the prompt prefix. **Fix:** Move `validation-state-injector.sh` from UserPromptSubmit to a `/status` slash command. CEO queries when they want it; it stops poisoning every prompt. 2 hours.

SECTION 4: WHAT'S MISSING (Gap Analysis)

These pieces do not exist and should:

G1 — Pre-dispatch task classifier hook

File: `~/.claude/hooks/pre-dispatch-classifier.sh` (does not exist) **What it does:** Takes task title + description, returns {company, agent, tier} before Task tool spawn. Currently John picks agents by intuition or uses generic `builder.md` catch-all. **Why it matters:** John dispatches 95%+ work via Task tool directly. Without classification, ZAKON #27 (one product per session) and company boundary rules have no enforcement point. **Spec:** Call `discover.js routing "$TASK_TITLE"`, parse JSON, inject into task metadata. If no match, route to Resolver. Block dispatch if company boundary violated.

G2 — Cost spike real-time alert

File: `~/system/daemons/cost-spike-alert.sh` (does not exist) **What it does:** Monitors `cost-tracker.js summary today` every 30 minutes. If hourly burn exceeds configurable threshold (e.g., $10K/hour based on verified units), sends immediate Slack DM to CEO. **Why it matters:** $129K/day with no revenue. A runaway Opus loop from midnight to 8am is currently invisible for 8 hours. The daily digest at 08:00 is 8 hours too late.

G3 — 3-vendor consensus router for /prompt-forge

File: `~/system/tools/consensus-router.js` (does not exist) **What it does:** For architecture decisions only (not routine tasks): fan out to Claude-Opus + GPT-5 + Gemini-2.5-Pro. A 4th cheap model judges divergence. 3-way agreement = ship. Divergent = human queue. **Why it matters:** The `feedback_john_recursive_drift.md` failure pattern — where John chains 8 nested MCs from a single strategic prompt — stems from single-model overconfidence. A 3-way disagreement is a natural circuit breaker. **Cost:** ~$0.50-2.00 per consensus call. Cheap insurance.

G4 — Daily CEO fleet brief via Slack

File: `~/system/tools/morning-fleet-brief.js` + `com.alai.morning-fleet-brief.plist` (do not exist) **What it does:** At 09:00 daily: daemon errors, cost yesterday vs prior day, MC status, service health, backup health, lumiscare.no status — 8 lines in #exec Slack. Current cost-daily-report writes to a file no one checks. **Why it matters:** CEO visibility gap. Kelsey's T5 brief showed every observability piece EXISTS but is NOT surfaced. The aggregation glue is missing.

G5 — Restore drill for mission-control.db

Documentation: `/Users/makinja/system/config/litestream-restore.yml` exists but no drill has been run. **What it does:** Monthly: `litestream restore -config ~/system/config/litestream.yml -replica mc-abs /tmp/mc-restore-test.db` then verify row count matches live DB. **Why it matters:** T2's single most dangerous gap: "If Litestream's Azure SP credentials have rotated or SP permissions have changed, the replicas may be stale without any alert." Without a drill, you do not know if your backup works until you need it.

G6 — Atomic task claim with BEGIN IMMEDIATE

File: `/Users/makinja/system/kernel/pi-orchestrator.js` (needs modification, not creation) **What it does:** Wraps 3-statement task claim in `db.transaction()`. Adds FK enforcement. Adds missing indexes. **Why it matters:** Race condition is latent. With pi-orchestrator currently idle, there is a window to fix this before resumption.

G7 — openai.js adapter + gpt-4o-mini Batch for RAG ingest

File: `~/system/tools/adapters/openai.js` (does not exist) **What it does:** Mirror of groq.js structure. Register in adapters/index.js. Wire lightrag-bulk-upload.js to call OpenAI Batch API for embeddings refresh. **Why it matters:** gpt-4o-mini is 5.3x cheaper input / 6.7x cheaper output than Haiku. For high-volume RAG ingest, this is the single largest unactualized cost reduction available.

G8 — WAL checkpoint cron

File: `~/system/tools/wal-checkpoint.js` + `com.alai.wal-checkpoint.plist` (do not exist) **What it does:** Every 4 hours: PRAGMA wal_checkpoint(TRUNCATE) on hivemind.db, flywheel.db, mission-control.db. Already described in T2 recommendations.

SECTION 5: DAY-TO-DAY OPERATIONS PLAYBOOK

This section is operational ritual, not strategy. Every command is concrete and runnable.

MORNING RITUAL (5 minutes, after boot, EVERY day)

```
# 1. Boot system (mandatory)
bash ~/system/boot.sh
```

2. CEO emails first (ZAKON)

node ~/system/tools/email-inbox.js pending

3. Cost check — first number you see

node ~/system/tools/cost-tracker.js summary today

Red flag: today > $50K (if units are dollars). Investigate immediately.

4. H-priority tasks

node ~/system/tools/mc.js list --owner john --priority H --status open

5. Daemon health

python3 -c " import json; d=json.load(open('/Users/makinja/system/state/daemon-fleet-status.json'))['daemons'] errs=[k for k,v in d.items() if 'err' in v['state'] or 'exit_512' in v['state']] print('Fleet errors:', len(errs)); [print(' -', e) for e in errs] "

Expected: 4 or fewer (the known 4). New errors = investigate before any work.

6. Backup health

```
tail -3 ~/system/logs/azure-db-backup.out tail -3 ~/system/logs/dr-sync.log
```

Expected: "COMPLETE" and "DR Sync COMPLETE" within last 24h / 6h

7. pi-orchestrator status

```
tail -2 ~/system/logs/pi-orchestrator.log
```

Expected: "No eligible tasks" (idle is OK) or recent cycle. ERROR = immediate triage.

8. Legal docs sync check (until fixed)

```
tail -3 ~/system/logs/legal-docs-azure-sync.log
```

Expected: "COMPLETE" or "All blobs uploaded". Until fixed: "FAIL" = escalate.

Total: 5 minutes. If any red flag: create MC before starting any other work.

DURING WORK: Pre-Dispatch Protocol

Before EVERY Task tool dispatch:

```
# Step 1: Route query
node ~/system/tools/discover.js routing "<task description in plain language>"
```

Returns company + agent. Use the named specialist, NOT generic builder.md

Step 2: If routing returns nothing, use Resolver

Dispatch to resolver.md with task description. Let it classify.

Step 3: Check current dispatch count

If you have dispatched 2+ tasks this session: confirm you are under cap.

Cap is 3 (current) — raise to 8 as per T3 recommendation.

Step 4: Mehanik clearance (H/BLOCKER only)

/mehanik — then wait for marker file before dispatch

Step 5: Dispatch with correct agent name

Task tool -> <named-expert>.md, NOT "builder"

Never dispatch to umbrella personas (codecraft.md, proveo.md, etc.) — they route ambiguously.

EVENING CHECK (3 minutes, end of work session)

```
# 1. Cost cumulative check
node ~/system/tools/cost-tracker.js summary today
```

Compare to morning. If increase > $30K during your session: check which tasks ran.

2. New daemon failures

python3 -c " import json; d=json.load(open('/Users/makinja/system/state/daemon-fleet-status.json'))['daemons'] errs=[k for k,v in d.items() if 'err' in v['state']] print('Errors:', len(errs)); [print(' -', e) for e in errs] "

3. Clear review queue

node ~/system/tools/mc.js list --status ready_for_review --json | python3 -c " import json,sys; tasks=json.load(sys.stdin) [print(f'#'+str(t['id']),t['title'][:60]) for t in tasks] "

Any tasks in ready_for_review owned by pi-orchestrator: these block pi-orch restart.

Tasks #10038, #10039, #10043 are current blockers.

4. Postflight compliance

node ~/system/tools/mc.js list --owner john --status done --json | tail -5

Confirm last 5 completed tasks have dod_evidence field populated.

WEEKLY REVIEW (30 minutes, every Monday)

```
# 1. Cost week-over-week
node ~/system/tools/cost-tracker.js summary week
```

Compare to prior week. Flag any >20% increase. Check model distribution — Opus >85% = review.

2. Full fleet error trending

Read ~/system/state/daemon-fleet-status.json summary block.

Errors should trend DOWN, not accumulate. Screenshot for comparison.

3. Azure backup verification

```
az storage blob list \ --account-name alaibackups0ebb \ --container-name system-db-backups \ --prefix "litestream/mission-control" \ --query "[?contains(name,'$(date +%Y-%m)')].{name:name,modified:properties.lastModified}" \ -o table
```

Expected: blobs from current week. None = Litestream silently

stopped.

4. Archive-first ledger sweep

cat /tmp/archive-first-scan-report-$(date +%Y%m%d).txt | head -20

Check candidate count. If growing: create MC to archive top-priority legal docs.

5. MC closure validation (ZAKON Hard Constraint #3)

node ~/system/tools/mc.js list --status done --owner john --json | \ python3 -c " import json,sys; tasks=json.load(sys.stdin) bad=[t for t in tasks if not t.get('dod_evidence') and not t.get('validation_timestamp')] print('Undocumented closures:', len(bad)) [print(' -', t['id'], t['title'][:50]) for t in bad[:5]] "

Any closures without evidence: investigate. MC #10173 failure pattern.

6. ZAKON #28 override count

grep "\[CEO_APPROVED\]" ~/.claude/hooks/john-max-depth-gate.log 2>/dev/null | \ grep "$(date +%Y-%m)" | wc -l

If >2/week: escalate to CEO. Override cadence is a leading indicator of drift.

7. Archive-first top candidates

node ~/system/tools/discover.js "unarchived legal docs contracts NDA"

Pick top 3, archive to archive.alai.no (ZAKON ARCHIVE FIRST)

MONTHLY ARCHITECTURE SWEEP (30 minutes, first of each month)

```
# 1. Full restore drill (THIS IS MANDATORY — never skip)
litestream restore \
  -config ~/system/config/litestream.yml \
  -replica mc-abs \
  /tmp/mc-restore-test.db
sqlite3 /tmp/mc-restore-test.db "SELECT COUNT(*) FROM tasks;"
sqlite3 ~/system/databases/mission-control.db "SELECT COUNT(*) FROM tasks;"
```

These numbers should match (within ~10 rows for replication lag). If not: ALERT.

rm /tmp/mc-restore-test.db

2. Agent inventory audit

ls ~/.claude/agents/*.md | wc -l jq '.mappings | length' ~/system/agents/specialist-mapping.json

Expected after cleanup: 22 files, 19 mappings. If growing without justification: review.

3. RAG/HiveMind growth

node ~/system/tools/cost-tracker.js summary month sqlite3 ~/system/databases/hivemind.db "SELECT COUNT(*) FROM intel;"

Track growth rate. If >50K new entries/month: verify source diversity.

4. Azure VM backup verification (manual)

ssh alai-admin@4.223.110.181 " cd /opt && du -sh bookstack-data/ vaultwarden-data/ 2>/dev/null

echo 'Last modified:' && ls -lt /opt/ | head -5 "

Until VM-level backup is configured: this is your only assurance.

5. Cost model distribution audit

node ~/system/tools/cost-tracker.js task-summary | head -20

Identify top 5 cost-generating tasks. Are they justified? Opus where Haiku sufficient?

6. Orphaned temp cleanup

ls /tmp/alai-azure-backup-* 2>/dev/null | wc -l

If >10: rm -rf /tmp/alai-azure-backup-* (stalled backup runs, ~700MB each)

ls /tmp/mehanik-cleared-* 2>/dev/null | wc -l

Clean stale mehanik marker files from completed sessions.

7. Litestream WAL checkpoint verification

sqlite3 ~/system/databases/hivemind.db "PRAGMA wal_checkpoint(TRUNCATE);" sqlite3 ~/system/databases/flywheel.db "PRAGMA wal_checkpoint(TRUNCATE);" sqlite3 ~/system/databases/mission-control.db "PRAGMA wal_checkpoint(TRUNCATE);" ls -lh ~/system/databases/*.db-wal

WAL files should be <10MB after checkpoint. If still large: investigate who holds long transactions.

SECTION 6: PRIORITIZED ROADMAP (Next 4 Weeks)

WEEK 1: P0 — Survival (Legal + DR + Observability + Data Integrity)

| Item | File | Effort | Success Metric | |------|------|--------|----------------| | Fix legal-docs-azure-sync.sh (--overwrite + ISBD path) | `/Users/makinja/system/daemons/legal-docs-azure-sync.sh` | 30 min | Next cron run completes with 84/84 blobs uploaded, exit 0 | | Verify cost units against Anthropic billing dashboard | Manual (console.anthropic.com vs cost-tracker.js) | 30 min | Reconciled dollar amount documented in MC | | Enable Azure VM-level backup policy | Azure Portal (manual) | 30 min | Azure Backup vault shows VM in protected state | | WAL checkpoint cron (4h interval) | `~/system/tools/wal-checkpoint.js` + plist (create new) | 2h | hivemind.db-wal < 10MB after first

run | | Reopen MC #10173: fix 4 remaining failures | `/Users/makinja/system/daemons/infra-drift-detector.sh` (git config renameLimit), investigate apply-knowledge + auto-verify-regression | 4h | fleet-status.json shows 0 errors for all 4 previously failing daemons | | Raise dispatch cap: 3 → 8 | `/Users/makinja/.claude/hooks/lock-john-dispatch-cap.sh` line 77 | 10 min | Session can dispatch 8 sub-agents without [CEO_APPROVED] | | Run first restore drill on mission-control.db | Manual (litestream restore command per playbook) | 30 min | Row count in /tmp restore matches live DB |

Week 1 total effort: ~8 hours

WEEK 2: P1 — Observability + Routing

| Item | File | Effort | Success Metric | |------|------|--------|----------------| | Morning fleet brief to Slack | `~/system/tools/morning-fleet-brief.js` + plist (new) | 2h | Daily 09:00 Slack message in #exec with 8-line brief | | Cost spike real-time alert | `~/system/daemons/cost-spike-alert.sh` + plist (new) | 2h | Alert fires when hourly burn > threshold; fires within 5min of trigger | | Fix ops-watchdog false alarms (--location flag) | `~/system/daemons/ops-watchdog.sh` | 30 min | docs.basicconsulting.no and vault.basicconsulting.no show green; lumiscare.no still red (correct) | | Repair specialist-mapping.json (17 orphans + 7 broken) | `~/system/agents/specialist-mapping.json` | 3h | `discover.js routing ""` returns results for all 40 agents; 0 broken entries | | Unblock pi-orchestrator: resolve tasks #10038, #10039, #10043 | Manual review of 3 tasks in ready_for_review | 2h | pi-orch log shows task pickup after unblock | | Clean cross-company-routes.json (remove HelixSupport, Proxima) | `~/system/config/cross-company-routes.json` | 30 min | No stale company references | | Bootstrap lightrag-auto-heal | Manual: launchctl bootstrap gui/$(id -u) ~/Library/LaunchAgents/com.alai.lightrag-auto-heal.plist | 5 min | launchctl list shows com.alai.lightrag-auto-heal with PID |

Week 2 total effort: ~10 hours

WEEK 3: P2 — Efficiency + Agent Hygiene

| Item | File | Effort | Success Metric | |------|------|--------|----------------| | Move validation-state-injector to /status | `~/.claude/hooks/validation-state-injector.sh` → rename to skill; remove from UserPromptSubmit in settings.json | 2h | UserPromptSubmit hooks: 6 → 4; prompt cache hit rate increases | | Retune mehanik.md: sonnet → haiku | `~/.claude/agents/mehanik.md` line 3 | 5 min | Pre-dispatch gate uses Haiku model; gate logic unchanged | | Delete 8 umbrella persona files | `~/.claude/agents/` agentforge.md, codecraft.md, finverge.md, flowforge.md, proveo.md, securion.md, skybound.md, vizu.md | 2h | discover.js routing forces resolution to named specialists; test 10 routing queries | | Consolidate 3 reviewers → 1 | gemini-reviewer.md, redzo-

reviewer.md, sentinel-validator.md → keep best, delete 2 | 1h | 1 review agent in ~/.claude/agents/; mapping updated | | Add openai.js adapter + OPENAI_API_KEY to Bitwarden | `~/system/tools/adapters/openai.js` (new) + register in adapters/index.js | 4h | comms-responder fallback chain includes OpenAI; confirmed via test call | | Wire lightrag-bulk-upload.js to gpt-4o-mini Batch | `~/system/tools/lightrag-bulk-upload.js` modification | 4h | Next bulk ingest run uses OpenAI Batch API; cost-tracker shows OpenAI spend | | Add FK enforcement to all DB connections | `pi-orchestrator.js`, `mc.js`, all tool scripts: add `db.pragma('foreign_keys = ON')` | 4h | PRAGMA foreign_key_check returns 0 violations on clean run | | Investigate builder.md + 4 persona ghosts with CEO | `~/.claude/agents/builder.md`, `alem-clone.md`, `dzevad-jahic.md`, `sylfest-lomheim.md` | 1h | Decision: delete or hard-scope each; documented in MC |

Week 3 total effort: ~18 hours

WEEK 4: P3 — Resilience + Architecture Foundation

| Item | File | Effort | Success Metric | |------|------|--------|----------------| | Atomic task claim (BEGIN IMMEDIATE) | `/Users/makinja/system/kernel/pi-orchestrator.js` getNextTask() function | 4h | 3-statement claim wrapped in db.transaction(); verified under load | | Add missing DB indexes | `pi-orchestrator.js` startup: idx_task_scheduling_next_eligible, idx_tasks_delegated_to | 2h | EXPLAIN QUERY PLAN on getNextTask() shows index use | | Monthly restore drill (recurring) | Manual (per Section 5 playbook) | 30 min | Row count verified; result documented in MC | | consensus-router.js POC for /prompt-forge | `~/system/tools/consensus-router.js` (new) | 8h | Claude-Opus + GPT-5 agree on a test architecture question; divergence creates human-queue task | | Fix Slack channel: dr-sync.sh → #exec | `~/system/tools/dr-sync.sh` | 5 min | No ERROR: Channel #ops not found in dr-sync log | | Add /tmp orphan cleanup to azure-db-backup.sh | `~/system/daemons/azure-db-backup.sh`: add trap cleanup EXIT | 1h | No /tmp/alai-azure-backup-* older than 24h | | Begin pi-orchestrator.js decomposition planning | Create BUILD-BLUEPRINT for 4-service split (task-classifier, model-router, agent-executor, task-scheduler) | 4h | Blueprint document; no code changes yet. Code split is a 2-4 week effort AFTER blueprint approved | | Connect flywheel.db distillation → OpenAI Stored Completions | `~/system/tools/distillation-exporter.js` (new): query used_for_training=0, send to OpenAI | 8h | First batch of traces submitted; gpt-4o-mini fine-tune job ID recorded |

Week 4 total effort: ~28 hours

SECTION 7: ARCHITECT'S VERDICT

Is ALAI salvageable? Yes, unambiguously. The foundational investment — Litestream replication with proper tier classification, the events.db transactional outbox schema, the circuit breaker pattern in task_scheduling, the ZAKON enforcement hook architecture, the Ollama local fleet with fine-tuned domain models — represents genuinely sophisticated engineering for a zero-revenue company. Most funded startups at this stage do not have backup infrastructure this well-designed.
What John must change about himself (process):
The core failure pattern is documented in `feedback_john_recursive_drift.md` and proven again in MC #10173: **John closes tasks before verifying outcomes.** Not once, not twice — this is a structural habit. An agent reports work done, John marks done, CEO finds it was not done. The architectural response (Hard Constraint #3, Proveo validation layer, evidence-gate hook) exists and is correct. But the constraint fires on mc.js done — which means John is still the gate, and John keeps bypassing it. The solution is not more process. It is personal discipline: never mark a task done without looking at a concrete output artifact. A task is done when you can point to the thing, not when the agent says it is done.

The second habit: John dispatches to generic agents (builder.md) instead of named specialists, and to umbrella personas (codecraft.md) instead of the actual expert. This is route-by-habit, not route-by-classification. The 17 orphaned agents and the routing system's 95% bypass rate are symptoms of the same thing.

What must change about the system (technical):
Three high-leverage changes: (1) WAL checkpoint cron — 2 hours of work prevents multi-minute data loss on restart. (2) Atomic task claim — 1 day of work eliminates the race condition before pi-orchestrator resumes. (3) Validation-state-injector out of UserPromptSubmit — 2 hours of work stops burning money on cache invalidation every prompt. These three changes alone are net-negative cost (they save more than they cost).

The pi-orchestrator.js monolith is the largest technical debt. 5,251 lines, 14 subsystems, single point of failure. It is not broken today — it is idle. But when it resumes at scale, the non-atomic claim and the 14-writer stampede will create problems that are hard to diagnose in production. Begin the decomposition design in week 4, execute the split over the following month.

On vendor lock-in: Do not migrate away from Claude Code as orchestrator. The switching cost is 10x the risk premium at current scale. DO add OpenAI for RAG ingest (gpt-4o-mini Batch) — it is the single highest-ROI cost reduction available without touching orchestration. DO verify the $129K/day number against actual billing before making any vendor decisions. If those numbers are real dollars, ALAI has a 30-day runway problem, not an architecture problem. **TL;DR for CEO:** The architecture is solid and salvageable — fix three things this week (legal docs sync, Azure VM backup, WAL checkpoint), raise the dispatch cap from 3 to 8, verify the cost numbers against real billing, and establish the habit of never marking a task done without a concrete artifact to point at.

Report authored by: Petter Graff, Chief Architect, T6 Master Synthesis Input: T1 (Resolver), T2 (Kleppmann), T3 (Anthropic), T4 (OpenAI), T5 (Kelsey Hightower) All findings tool-verified by source teams. Zero memory assertions. Date: 2026-04-30

Revision #2
Created 2026-05-01 08:08:27 UTC by John
Updated 2026-06-07 20:00:42 UTC by John