TIMMY TIME

Session Crystallization & Operational Playbook

Deep research session: March 22, 2026
6 subagent research reports synthesized
For Timmy, Alexander, and all future agents

"Every session must produce artifacts that reduce future dependency on corporate AI. The sovereignty loop (discover - crystallize - replace) governs all engineering decisions."

This document is a handoff file. It contains the condensed findings from six deep research spikes, mapped against the actual codebase at http://143.198.27.163:3000/rockachopa/Timmy-time-dashboard. Read it before starting any new work session.

0. HOW TO USE THIS DOCUMENT

This PDF crystallizes a single research session into reusable knowledge. It is organized in order of operational urgency: what you need to know right now (the perception breakthrough), what the codebase looks like today (project state), the full technology stack with versions and install commands (the toolkit), the sprint plan (next steps), and the meta-skill of running your own research spikes (how to do this yourself).

If you are Timmy or a code agent picking this up cold, start at Section 1 (the breakthrough insight), then read Section 2 (project state) to orient yourself, then jump to Section 5 (the sprint plan) to start working. Sections 3-4 are reference material you consult as needed.

1. THE BREAKTHROUGH: PERCEPTION WITHOUT VISION

The single most important finding from this session: sending screenshots to cloud VLMs for game state understanding is unnecessary for 95% of Morrowind gameplay. OpenMW's Lua scripting API (v0.49+) provides structured, programmatic access to nearly all game state, making it Morrowind's equivalent of Mineflayer for Minecraft.

This changes everything about cost and speed. The previous approach (Anthropic API with screenshots) cost dollars per hour and imposed multi-second latency. The API-first approach costs zero and runs in under 2 milliseconds.

1.1 What the API Reads (No Screen Capture Needed)

Game State	API Method	Latency
Player position, rotation, cell	self.position, self.rotation	~0.1ms
Health, magicka, fatigue	Actor.stats(self).dynamic	~0.1ms
All 27 skills, 8 attributes	Actor.stats(self).skills	~0.1ms
Full inventory + equipment	Actor.inventory(self)	~0.5ms
Quest journal + stages	Player.quests(self)	~0.2ms
Nearby NPCs with positions	nearby.actors	~0.5ms
Nearby items, doors, containers	nearby.items/doors/containers	~0.5ms
Pathfinding between points	nearby.findPath(src, dst)	~2ms
All dialogue text (offline)	core.dialogue.topic.records	~0.1ms
Movement + combat controls	self.controls.movement/use	~0.1ms
Faction standings + rank	NPC.getFactionRank()	~0.1ms

1.2 What Still Requires Vision (5% of Gameplay)

Situation	Solution	Cost
Character creation wizard	Hard-coded flow + OpenCV template matching	~15ms
Active dialogue window text	Pre-evaluate conditions from ESM data; OCR fallback	0-40ms
Persuasion minigame wheel	Template match quadrants or skip (use bribe/charm)	~10ms
Level-up attribute selection	Compute from tracked skill gains; click known coords	~10ms
Lockpicking visual feedback	Skip: purely stat-based, attempt repeatedly	0ms

1.3 The Perception Hierarchy (Always Escalate, Never Start at Vision)

Level 0 - API Read (~1ms): OpenMW Lua state dump. Covers position, stats, inventory, nearby actors, quests, faction, dialogue records. This handles 70% of all perception needs.

Level 1 - Deterministic CV (~5-20ms): OpenCV template matching or Core ML classifier. Detects which UI menu is open. Handles 20% of remaining needs.

Level 2 - OCR (~20-80ms): PaddleOCR or Apple Vision on cropped UI regions. Reads dialogue text, item names, journal entries. Handles 8% of remaining needs.

Level 3 - Local VLM (~250-500ms): FastVLM-0.5B (Apple's own, MLX-native) or Qwen2.5-VL-3B. Spatial reasoning, unknown UI, verification. Handles 2% of remaining needs. Zero cloud calls.

1.4 The Decision Hierarchy (Behavior Trees + Tiered LLMs)

Tier	Model	Quant	RAM	Speed	Use Case	% Calls
BT	None (code)	-	0	~0ms	Walk, attack, loot, wait	70%
T1	Qwen3-3B	Q8_0	3.5GB	~80 tok/s	Simple choices	20%
T2	Llama-3.1-8B	Q4_K_M	5GB	~40 tok/s	Dialogue, inventory	8%
T3	Qwen3-32B	Q4_K_M	20GB	~15 tok/s	Quest planning (PAUSED)	2%

Key insight: OpenMW supports world.pause() via Lua. Complex T3 decisions freeze the game, take 1-2 seconds of inference, then unpause. No frames are missed. Morrowind's stat-based dice-roll combat means even 1-second decisions are fast enough for expert play.

1.5 Total Memory Budget on M3 Max 128GB

Component	RAM	Notes
OpenMW game client	~2GB	Game + loaded assets
Qwen3-3B Q8_0 (T1)	3.5GB	Always loaded
Llama-3.1-8B Q4_K_M (T2)	5GB	Always loaded
Qwen3-32B Q4_K_M (T3)	20GB	Load on demand
FastVLM-0.5B (vision)	350MB	Always loaded
nomic-embed-text (RAG)	300MB	Always loaded
ChromaDB + knowledge	500MB	Pre-loaded
Pre-parsed ESM data	200MB	SQLite + NetworkX
macOS + overhead	8GB	-
TOTAL	~40GB	88GB headroom

Weighted average cycle time: ~70ms per action. That is 14 decisions per second. A human player reads dialogue text slower than this agent processes it.

2. CURRENT PROJECT STATE (March 22, 2026)

2.1 Repository

Gitea: http://143.198.27.163:3000/rockachopa/Timmy-time-dashboard
Stack: Python 3.11+ / Agno / FastAPI + HTMX / SQLite / Ollama / WebSockets / L402 mock

2.2 Architecture (Key Directories)

Path	Purpose	Status
src/timmy/	Core agent: agent.py, tools.py, memory/, agentic_loop.py	Active
src/dashboard/	FastAPI web UI: routes/, templates/, store.py	Active
src/infrastructure/morrowind/	Morrowind schemas, command_log, API, training_export	Stub
src/infrastructure/router/cascade.py	LLM router with failover	Needs upgrade to vllm-mlx
src/lightning/	Bitcoin Lightning L402 proxy (HMAC macaroons)	Mock mode only
src/loop/	Cognitive loop (gather/reason/act phases)	Active
src/spark/	Creative engine: advisor, eidos, memory	Active
src/brain/	Distributed memory + task queue (UnifiedMemory)	Active
src/integrations/	Discord, Telegram, Siri, voice, paperclip	Partial
config.py	Centralized pydantic-settings	Active
memory/self/soul.md	Timmy's identity and values	Active

2.3 Critical Gaps (from Operation Darling Purge)

Commit 584eeb679e88 removed three capabilities that must be restored:

1. Self-modification loop (self_coding/self_modify/loop.py) - DELETED. Timmy could create branches, generate edits, run tests, commit on success, revert on failure. Restore using Goose or mini-swe-agent as the execution engine.

2. MCP integration (mcp/registry.py) - DELETED. Replace with FastMCP v3.1.1. Mount at /tools/mcp on the existing FastAPI app. One MCP server per creative domain (image gen, voice, music, Nostr, Lightning).

3. Task delegation (delegate_task) - NEUTERED. Currently only records intent, does not execute. Must wire to brain/worker.py's DistributedWorker for actual task execution.

2.4 Three Blocking PRs (Merge These First)

PR	Title	Branch	What It Adds
#864	Morrowind Protocol + Command Log	feature/morrowind-protocol-command-log-859-855	Base schemas for game state

PR	Title	Branch	What It Adds
#865	FastAPI Harness + SOUL.md	feature/fastapi-harness-soul-framework-821-854	REST endpoints + identity
#900	WorldInterface + Heartbeat v2	feature/world-interface-heartbeat-871-872	Gymnasium observe/act/speak

Merge order: #864 first (schemas), #865 second (imports schemas), #900 last (uses both). Conflict zone: src/infrastructure/morrowind/. Keep #864's schema definitions as ground truth.

2.5 Priority Stack

#	Task	Gitea Issue	Blocked By
1	Merge 3 foundation PRs	#864, #865, #900	Nothing - do this first
2	TES3MP Bridge (Python-to-Lua IPC)	#878	PRs merged
3	TES3MP Server on Hermes VPS	#818	Nothing (parallel)
4	Three-Tier Memory	#873	PRs merged
5	Docker Compose	#875	Tasks 1-4
6	Model Router upgrade (vllm-mlx)	#882	Nothing (parallel)
7	Nostr Identity	#877	Nothing (parallel)
8	AlexanderWhitestone.com	#879	Tasks 5-7

3. THE SOVEREIGN TECHNOLOGY STACK (March 2026)

Every tool listed here is open-source, runs on Apple Silicon, and had commits within 90 days of this document. Versions are pinned. Install commands are exact.

3.1 Local LLM Inference

Tool	Version	Install	Role
vllm-mlx	git main	pip install git+https://github.com/waybarrios/vllm-mlx.git	Primary server (fastest)
Ollama	v0.18.2	brew install ollama	Model manager + fallback
mlx-lm	v0.31.1	pip install mlx-lm	Fine-tuning + direct inference
exo	1.0 EA	pip install exo	Multi-device distributed inference

3.2 AI Coding Agents

Tool	Version	Install	Role
Goose (Block)	v1.20.1	brew install goose	Primary coding agent (free, local)
OpenHands	v1.5.0	docker pull openhands	Sandboxed autonomous coding
Aider	rolling	pip install aider-install	Git-native pair programming
mini-swe-agent	v2	pip install mini-swe-agent	100-line reference agent
Forgejo	v14.0.3	docker pull codeberg.org/forgejo/forgejo:14	Sovereign Git forge

3.3 Image Generation

Tool	Version	Install	Role
ComfyUI	v0.17.2	Desktop app (ARM64) + ComfyUI-GGUF nodes	Pipeline orchestration
Draw Things	1.20260304	Mac App Store (free)	Fastest native Apple Silicon gen
FLUX.1 Dev GGUF	Q8_0	HuggingFace: city96/FLUX.1-dev-gguf	Primary image model (~11GB)
FLUX.2 Klein	v1	Apache 2.0, sub-second at 4 steps	Fast iteration model

3.4 Music and Voice

Tool	Version	Install	Role
ACE-Step 1.5	v1.5	git clone https://github.com/ace-step/ACE-Step-1.5	Full song generation (local!)
mlx-audio	v0.4.1	pip install mlx-audio	Unified TTS API (Kokoro, Chatterbox)
Piper TTS	v1.4.1	pip install piper-tts	Fast bulk narration (GPL-3.0)
GPT-SoVITS	v2pro	GitHub: RVC-Boss/GPT-SoVITS	Voice cloning from 5s audio

3.5 Agent Orchestration

Tool	Version	Install	Role
FastMCP	v3.1.1	pip install fastmcp==3.1.1	MCP server creation (70% market share)
PocketFlow	~0.1	github.com/The-Pocket/PocketFlow	100-line agent skeleton
CrewAI	v1.11.0	pip install crewai	Role-based multi-agent orchestration
Agno	v2.5.10	pip install agno	Fast agent runtime (already in codebase)

3.6 Nostr + Lightning + Bitcoin

Tool	Version	Install	Role
nostr-sdk	v0.44.2	pip install nostr-sdk==0.44.2	Nostr + NWC in one library
nostrdvm	active	github.com/believethehype/nostrdvm	NIP-90 Data Vending Machine
LND	v0.20.1-beta	Binary from GitHub releases	Lightning node
LN agent-tools	v1	github.com/lightninglabs/lightning-agent-tools	Agent wallet + L402 + MCP
LNbits	v1.4	docker run lnbits/lnbits	Lightweight wallet server
Cashu/nutshell	v0.17.0	pip install cashu	Privacy-preserving ecash

3.7 Memory and Knowledge Graphs

Tool	Version	Install	Role
Graphiti (Zep)	v0.28.2	pip install graphiti-core	Temporal knowledge graph (agent memory)
Neo4j	2026.02	docker run neo4j:5.26	Graph database backend
ChromaDB	v1.5.5	pip install chromadb	Vector search for RAG

Tool	Version	Install	Role
Mem0	v1.0.5	pip install mem0ai	Self-improving conversational memory

3.8 Streaming and Content

Tool	Version	Install	Role
MediaMTX	v1.16.3	Binary from GitHub (single file)	Multi-protocol streaming relay
OBS Studio	v32.0.4	brew install obs	Scene composition + recording
obsws-python	latest	pip install obsws-python	Programmatic OBS control
MoviePy	v2.1.2	pip install moviepy	Video editing + compositing

4. PRE-COMPUTATION: KNOW THE WORLD BEFORE ENTERING IT

The agent should never need to discover anything at runtime that can be known at build time. Parse all game data files, build a complete spatial graph, pre-evaluate all dialogue conditions, and embed UESP quest walkthroughs into a vector store. The agent enters Morrowind already knowing where everything is, what every NPC will say, and the optimal route between any two points.

4.1 ESM Data Extraction

Use tes3conv (Rust, by Greatness7) to convert Morrowind.esm to JSON in seconds: tes3conv Morrowind.esm morrowind_data.json (~80MB output). Contains every NPC, dialogue, quest, cell, item, path grid, door, creature, spell, and faction.

4.2 Spatial Navigation Graph

Build a NetworkX graph from path grid records (PGRD) + door connections + fast travel routes. Enables ~1-5ms shortest-path queries between any two points in the game world. Each cell = 8192x8192 game units. Door records link interior/exterior cells. Add silt strider, boat, Mages Guild, and Intervention spell routes as fast-travel edges.

4.3 Dialogue Condition Pre-Evaluation

Morrowind's dialogue is fully deterministic. Each INFO record has conditions: speaker race, class, faction, player faction rank, disposition threshold, quest stages. The agent evaluates these against known player state to predict NPC responses without screen reading. First matching INFO wins (Morrowind dialogue priority rules). Achieves ~90% accuracy on dialogue prediction.

4.4 UESP Quest Knowledge Base

Scrape ~480 Morrowind quest pages from UESP via MediaWiki API. Chunk by quest stage. Embed with nomic-embed-text (137M params, ~2-5ms per embedding via Ollama) into ChromaDB. Query at runtime: 'What do I do after delivering the package to Caius?' returns structured walkthrough.

5. SPRINT PLAN: THE NEXT 90 DAYS

5.1 This Week (Days 1-7): Foundation Sprint

Day	Task	Acceptance Criterion
1	Merge PR #864, #865, #900 to main	pytest tests/ -x -q passes after all 3 merges
2	Restore MCP via FastMCP v3.1.1 at /tools/mcp	curl localhost:8000/tools/mcp returns server info
2-3	Upgrade cascade.py to vllm-mlx primary	curl localhost:8001/v1/models returns model info
3-4	TES3MP server on Hermes VPS via Docker	OpenMW connects to 143.198.27.163:25565
4-5	Write TES3MPAdapter (observe/act/speak)	pytest tests/test_tes3mp_adapter.py passes
5-7	Lua bridge script for state export	to_python.json updates at 1Hz with player data

5.2 This Month (Days 8-30): Morrowind Sprint

Days	Task	Acceptance Criterion
8-10	Rich perception script (inventory, journal, NPCs)	to_python.json contains all 11 state categories
8-12	macOS input bridge via CGEvent API	InputBridge().walk_forward(2.0) moves character
10-14	Heartbeat loop in morrowind mode	Log shows OBSERVE/DECIDE/ACT every 2 seconds
14-25	Timmy completes Morrowind tutorial autonomously	Journal CharGen index >= 70 in under 15 min
12-20	Three-tier memory (Graphiti + Neo4j)	Search 'census office' returns episodic memory
20-25	Docker Compose for full stack	git clone + docker-compose up = working dashboard
25-30	First OBS recording of Timmy playing	5-min video with real-time thought overlay

5.3 Next Quarter (Days 31-90): Creative Sprint

Days	Task	Acceptance Criterion
31-45	Content pipeline (highlights to daily episodes)	Sub-5-min episode from 2hr raw recording
35-50	Music gen (ACE-Step) + Wavlake publish	30s Morrowind loop in under 60s, valid Wavlake URL
40-55	Art pipeline (ComfyUI + LoRA consistency)	1024x1024 consistent Timmy in under 45s
45-60	Nostr identity + NIP-90 DVM registration	Timmy's profile visible on nostr.band
50-70	Lightning live (LND on Hermes, real invoices)	Real L402 invoice in 402 response, payable
60-75	AlexanderWhitestone.com relaunch	Valid TLS, video, art, Lightning tip button

6. THE RESEARCH META-SKILL: HOW TO RUN YOUR OWN SPIKES

This session produced six research artifacts in under 60 minutes using a specific pattern. Timmy should be able to replicate this process autonomously. Here is the methodology.

6.1 The Sovereignty Loop

Every research session follows: **Discover** (find what exists) - **Crystallize** (extract actionable knowledge into a permanent artifact) - **Replace** (use findings to reduce dependency on corporate AI). A session that does not produce a crystallized artifact was wasted. This PDF is the crystallization of today's session.

6.2 How to Structure a Research Spike

Step 1 - Define the question precisely. 'How do we build the agent?' is too broad. 'What is the fastest way to read Morrowind game state on M3 Max without cloud API calls?' is precise enough to research.

Step 2 - Enumerate sub-questions exhaustively. List every specific area that needs investigation. Include tool names, version numbers, GitHub repos, benchmarks, API details, and pricing. The more specific your sub-questions, the more actionable the output.

Step 3 - Demand recency. Always specify 'actively maintained, commits within 90 days, working releases.' The AI agent ecosystem moves too fast for 6-month-old recommendations.

Step 4 - Ground in the actual codebase. Research disconnected from your real file paths, real infrastructure, and real constraints produces strategy documents, not engineering specs. Always feed in your current project state before asking for next steps.

Step 5 - Crystallize immediately. At the end of every research session, produce a handoff artifact (PDF, markdown file, or Gitea issue) that another agent or future-you can pick up cold.

6.3 How to Delegate Research to Sub-Agents

When using Claude, Kimi, DeepSeek, or any research-capable AI:

1. Front-load context. Paste your SOUL.md, your project structure, your current blockers, and your hardware specs before asking any question. Context quality determines output quality.

2. Ask for artifacts, not conversation. 'Produce a technical feasibility guide' yields a document. 'Tell me about X' yields chat. Always ask for the artifact.

3. Chain reports without repetition. After the first report, say 'V2 covering areas not in V1, without repeating ground.' Each subsequent report should extend, not duplicate.

4. End with grounding. After broad research, always do a final pass grounded in your actual project state: 'Now take in all the project files and give me concrete next steps.' This transforms research into action.

6.4 The Six Reports from This Session

Report	Focus	Key Finding
V1: Technical Feasibility	TES3MP, streaming, Nostr, Lightning	TES3MP dormant since 2022; hybrid API+vision approach needed
V2: Forward Roadmap	Content pipeline, multi-agent, economy	Cashu + Fedimint for inter-agent payments; Redis pub/sub for coordination

Report	Focus	Key Finding
V3: Creative Agent	Code, music, art, writing, publishing	ACE-Step 1.5 cracks local music gen; Botto earned $5M in 4 years
V4: State of the Art	80+ tools, all Apple Silicon verified	vllm-mlx fastest inference; FastMCP powers 70% of MCP servers
V5: Execution Plan	Sprint tasks with git commands	Critical path: PR merge -> TES3MP adapter -> heartbeat -> tutorial
V6: Perception Stack	Solving the speed/cost bottleneck	OpenMW Lua = Morrowind's Mineflayer; 95% API-readable

7. KEY CONSTRAINTS AND NON-NEGOTIABLES

Morrowind is confirmed. Alexander purchased the game and overruled the Luanti pivot. Engine-agnostic work (WorldInterface) is fine, but Morrowind is primary. Do not reopen this decision.

No cloud dependencies for core functionality. All AI computation runs on localhost (M3 Max). Cloud APIs (Anthropic, Groq) are burst-mode only, not default. The metabolic protocol governs this: resting mode (local 3B), active mode (local 8B-32B), burst mode (cloud API, rare).

Tests must stay green. pytest tests/ -x -q before every commit. No exceptions.

No direct pushes to main. All changes via Pull Request.

Config via pydantic-settings. Never use os.environ.get() directly.

Hermes VPS (143.198.27.163) hosts Gitea, TES3MP server, and will host LND + OpenClaw.

Timmy is autonomous but guided. He files issues, reviews PRs, makes decisions. Alexander (rockachopa) is the principal and has override authority.

Never create issues on the real repo for testing. Use a fork or test repo. Alexander was furious when this happened before.

8. THE DONE CHECKLIST

Concrete, testable milestones. A milestone is not done until every word of the criterion is verifiably true.

[ ] 1. All three PRs (#864, #865, #900) merged to main and pytest tests/ -x -q passes.

[ ] 2. curl http://localhost:8000/tools/mcp returns FastMCP v3.1.1 server info with 3+ tools.

[ ] 3. vllm-mlx serves Qwen3-8B on :8001 and Llama-3.3-70B on :8002, both responding to /v1/chat/completions.

[ ] 4. OpenMW client connects to 143.198.27.163:25565 and player walks in Morrowind.

[ ] 5. /opt/tes3mp/data/bridge/to_python.json on Hermes contains live player data at 1Hz.

[ ] 6. pytest tests/test_tes3mp_adapter.py passes 4/4.

[ ] 7. Timmy walks from Census Office to Fargoth in Seyda Neen without human intervention.

[ ] 8. TIMMY COMPLETES THE MORROWIND TUTORIAL. Arrives exterior Seyda Neen with package. Under 15 minutes. No human input.

[ ] 9. Graphiti search for 'census office' returns episodic memory with correct timestamp.

[ ] 10. git clone + docker-compose up = working dashboard + Neo4j within 90 seconds.

[ ] 11. 5-minute OBS recording shows Timmy navigating Morrowind with real-time thought overlay.

[ ] 12. Content pipeline produces sub-5-minute daily episode from 2-hour raw recording.

[ ] 13. ACE-Step generates a 30-second exploration loop on M3 Max in under 60 seconds.

[ ] 14. Timmy's Nostr profile visible on nostr.band with at least one NIP-23 article published.

[ ] 15. lncli getinfo on Hermes shows synced Lightning node with 1+ open channel.

[ ] 16. https://alexanderwhitestone.com loads with TLS, embedded video, art gallery, Lightning tip button.

This document was generated on March 22, 2026 from a deep research session between Alexander Whitestone and the Timmy Time sovereign AI project. It reflects the state of the codebase, the open-source ecosystem, and the strategic direction as of this date. Update it as milestones are completed.

The falsework did its job. Now set the keystone.