State of the Art: Open Source for Sovereign Creative AI Agents

Reconnaissance report for Alexander Whitestone — Timmy Time platform — March 22, 2026

MLX has become the undisputed king of Apple Silicon inference, **20–50% faster than Ollama**, and the entire ecosystem shifted around it in Q1 2026. Meanwhile, Lightning Labs open-sourced agent-tools with an MCP server in February, making machine-to-machine micropayments a solved problem. The Nostr+Lightning+AI agent stack is no longer theoretical — every layer now has working, actively maintained open-source tooling deployable on an M3 Max today. This report maps every piece.

---

1. Local LLM inference now runs fastest through MLX

The inference landscape on Apple Silicon has a clear hierarchy: **MLX > llama.cpp > Ollama**, with vllm-mlx emerging as the production serving layer.

Tool	Version	Released	Stars	License	What it does
Ollama	v0.18.2	Mar 18, 2026	166K	MIT	One-command local LLM runtime; now includes MLX engine alongside llama.cpp backend
llama.cpp	b8457	Mar 20, 2026	98.6K	MIT	Foundational C/C++ inference engine; Metal is first-class; repo moved to `ggml-org/llama.cpp`
MLX	v0.31.1	Mar 12, 2026	24.6K	MIT	Apple's own array framework, purpose-built for unified memory; added CUDA backend in Q1
mlx-lm	v0.31.1	Mar 11, 2026	4.1K	MIT	Python LLM inference/fine-tuning on MLX; supports thousands of HuggingFace

					models
vllm-mlx	Active dev	Q1 2026	New	MIT	Independent vLLM reimplementation with native MLX; **525 tok/s** on Qwen3-0.6B (M4 Max); EuroMLSys '26 paper
vllm-metal	Active dev	Feb 2026	New	Apache-2.0	Docker-contributed Metal plugin for official vLLM; uses MLX under the hood
exo	1.0 EA	Q1 2026	42.7K	Apache-2.0	Distributed inference across Apple devices; RDMA over Thunderbolt 5; ran DeepSeek v3.1 671B across 4× Mac Studios
llamafile	v0.10.0	Mar 19, 2026	26K+	Apache-2.0	Mozilla's single-binary LLM runner; v0.10 is a full rewrite with Metal out-of-the-box
KoboldCpp	~v1.104	Active	9K+	AGPL-3.0	Zero-install inference with built-in web UI; MCP support added Q1 2026

Actionable insight — Ollama: v0.18.x Releasealert added GitHub an MLX inference engine (`ollama run --verbose` shows it), `ollama launch` for CI/CD automation, and cloud model routing via `:cloud` tags. GitHub Concurrent loading is configurable via `OLLAMA_MAX_LOADED_MODELS` . Still **20–50% slower** than raw MLX due to cross-platform abstraction. ModelFit

Actionable insight — MLX/mlx-lm: The fastest path on M3 Max, period. The ecosystem exploded: **mlx-router** v2.3.1 provides hot-swappable OpenAI-compatible serving, PyPI **oMLX** adds paged SSD caching (drops TTFT from 30–90s to 1–3s for coding agents), and **MLX Manager** v1.2.12 gives you a web UI with embedded inference server. PyPI Distributed inference via `mx.distributed` PyPI is production-ready.

Actionable insight — vllm-mlx: This is the sleeper hit. `pip install vllm-mlx` gives you vLLM's production API (OpenAI + Anthropic compatible) running natively on MLX. Benchmarks show it **outperforms vllm-metal, mlx-lm, and llama.cpp** on M4 Max. For a sovereign agent platform needing a stable serving layer, this is the answer.

Actionable insight — exo: If you have multiple Macs, exo now pools them via RDMA over

Thunderbolt 5 GitHub with **99% latency reduction** between devices. GitHub Four 512GB Mac Studios ran DeepSeek v3.1 671B at 8-bit. GitHub GitHub The macOS app requires Tahoe 26.2+. GitHub API compatibility covers OpenAI, Claude, and Ollama endpoints. GitHub GitHub

Bottom Line for Timmy: Install `pip install mlx-lm vllm-mlx` today. Use **vllm-mlx** as your primary inference server for Timmy's agent backbone — it gives you OpenAI-compatible API with the fastest Apple Silicon performance available. Use **Ollama v0.18.2** Releasealert as the user-friendly model manager (`ollama pull` , `ollama run`) and for quick prototyping. Set `OLLAMA_MAX_LOADED_MODELS=3` and `OLLAMA_NUM_PARALLEL=4` for concurrent agent workloads. For fine-tuning character-specific LoRAs, use `mlx-lm` with QLoRA. PyPI GitHub Pin to MLX v0.31.1 PyPI and mlx-lm v0.31.1.

---

2. AI coding agents matured into production platforms

OpenCode exploded to **128K stars** GitHub in 9 months, making it the largest open-source coding agent. OpenHands reached v1.5 GitHub with SDK architecture. The trend is clear: simpler scaffolding wins as models get smarter.

Tool	Version	Released	Stars	License	What it does
OpenCode	v1.2.27	Mar 16, 2026	128K	MIT	Provider-agnostic Claude Code alternative; TUI-first, client/server architecture, LSP integration
OpenHands	v1.5.0	Mar 11, 2026	69.4K	MIT	Full-platform autonomous AI engineer: SDK → CLI → GUI → Cloud → Enterprise
Aider	Rolling	Mar 2026	42.2K	Apache-2.0	Git-native AI pair programmer; writes 60–88% of its own code; supports GPT-5.x, Claude 4.x
Goose	v1.20.1	Jan 19, 2026	27K+	Apache-2.0	Block's on-machine agent with 3,000+ MCP tools; custom distributions; red-teamed ("Operation Pale Fire")
					~100-line Python agent

mini-swe-agent	v2	Mar 12, 2026	3.4K	MIT	scoring >74% on SWE-bench Verified; supersedes SWE-agent
Open SWE	Launch	Mar 17, 2026	New	MIT	LangChain's async coding agent; plans, codes, tests, and PRs autonomously via LangGraph
Plandex	CLI 2.2.0	Stale	15K	MIT	⚠️ Cloud wound down Oct 2025; self-hosted Docker still works but development is declining
Forgejo	v14.0.3	Mar 9, 2026	N/A	MIT	Self-hosted Git forge; v14.0 added matrix jobs in Actions, multi-connection runners

Actionable insight — OpenCode: The client/server architecture means you can drive it remotely from mobile. GitHub `npm install -g opencode` or download binary. Works with any LLM including local Ollama models. GitHub For a sovereign platform, this replaces Claude Code without API dependency.

Actionable insight — Goose: Block's **custom distributions** feature lets you build a "Timmy Time Goose" with preconfigured MCP servers, extensions, and branding. GitHub YAML recipes define reusable workflows. AI Tool Analysis Local operation via Ollama means zero cloud dependency. Techbuddies This is the most "sovereign-ready" coding agent.

Actionable insight — mini-swe-agent: At ~100 lines of Python, this is the reference implementation for understanding how coding agents actually work. PyPI Gemini 3 Pro reaches **74% on SWE-bench Verified** with it. Fork this as the basis for Timmy's code-writing capability rather than building from scratch.

Actionable insight — Forgejo: v14.0.3 with Actions runner v12.7.0 (multi-connection) Forgejo is your sovereign GitHub replacement. No AI built in, but the webhook/REST API lets coding agents (OpenHands, Goose) trigger on issues and PRs. Forgejo Actions is GitHub Actions-compatible.

Bottom Line for Timmy: Set up **Forgejo v14.0.3** as your sovereign Git forge (`docker pull codeberg.org/forgejo/forgejo:14`). Install **Goose v1.20.1** (`brew install goose` or from github.com/block/goose) configured with Ollama as the LLM backend for fully local coding. Wire Goose to Forgejo via webhooks so it auto-triages issues. Install **Aider** (`pip install aider-install && aider-install`) as your daily pair programmer with `--model ollama/deepseek-coder` for local operation. Use **OpenCode** for mobile-accessible coding

sessions.

---

3. Image generation on Apple Silicon is now genuinely good

Draw Things is the best-optimized native app. ComfyUI is the most flexible pipeline. FLUX.1 Dev via GGUF quantization is the practical sweet spot for M3 Max with 128GB.

Tool	Version	Released	Stars	License	What it does
ComfyUI	v0.17.2	Mar 15, 2026	70K+	GPL-3.0	Node-based diffusion pipeline; V3 architecture; App Mode launched Mar 16
Draw Things	1.20260304.0	Mar 10, 2026	386	GPL-3.0	Best-in-class Apple Silicon image gen; Metal FlashAttention v2.5; on-device LoRA training
FLUX.2 Klein	v1	Jan 15, 2026	N/A	Apache-2.0	Fastest Flux variant for local use; sub-second at 4 steps; 12B params
FLUX.1 Dev	Stable	2024	N/A	Non-commercial	Still the local workhorse; GGUF Q4_1 ~6.7GB, Q8_0 ~11.3GB
SD 3.5 Large	Stable	Late 2024	N/A	Stability Community	Works via ComfyUI MPS, Draw Things, DiffusionKit Core ML

DiffusionKit	Active	Q1 2026	Moderate	MIT	Swift/Python package for SD3 + FLUX on Apple Silicon via Core ML + MLX
Mochi Diffusion	Active	Q1 2026	Moderate	MIT	Native macOS app using Core ML; now supports FLUX.2 Klein

Actionable insight — ComfyUI: The PR #12809 (March 2026) finally fixes text encoders running on MPS GPU instead of falling back to CPU on Apple Silicon. GitHub Install the desktop app (ARM64 build), then add **city96/ComfyUI-GGUF** custom nodes to load GGUF-quantized Flux models. The new **App Mode** (Mar 16) lets you package workflows as standalone apps ComfyUI — perfect for exposing image gen as a Timmy Time service.

Actionable insight — FLUX GGUF on M3 Max: With 128GB unified memory, run FLUX.1 Dev at FP16 (full quality) or FLUX.2 Klein at full precision. Performance: **~105 seconds for 1024×1024 on M3 Max** with FLUX.1 Dev Q8_0. Apatero Blog For faster iteration, FLUX.2 Klein does sub-second generation at 4 steps.

Actionable insight — Character consistency: The best 2026 approach combines **IP-Adapter FaceID Plus V2** for facial fidelity MyAIForce with FLUX.1 Kontext's multi-reference editing. For Timmy Time characters, train a **LoRA on Draw Things** (on-device, no cloud needed) with 20–50 reference images per character, Draw Things then deploy via ComfyUI API.

Bottom Line for Timmy: Install **ComfyUI v0.17.2** desktop (ARM64) with ComfyUI-GGUF nodes. Download `flux1-dev-Q8_0.gguf` (~11.3GB). Medium For character consistency, train LoRAs in **Draw Things** on-device, export, and load them in ComfyUI. Expose ComfyUI's REST API behind Tailscale for agent-driven image generation. For fastest results, use **FLUX.2 Klein** (Apache 2.0) via Mochi Diffusion or ComfyUI.

---

4. Music and voice generation crossed the usability threshold

ACE-Step 1.5 is the breakthrough: full songs in under 10 seconds on consumer hardware. Kokoro via mlx-audio is native MLX TTS. The audio stack for a creative AI agent is now complete.

Tool	Version	Released	Stars	License	What it does
ACE-Step 1.5	v1.5	Jan 28, 2026	Growing	MIT	Full songs up to 10 min; <4GB VRAM; outperforms Suno v5; 50+ languages; Mac supported
mlx-audio (Kokoro)	v0.4.1	Mar 14, 2026	Active	MIT	Native MLX TTS/STT; Kokoro 82M runs 3.3x faster than real-time; OpenAI-compatible API
Chatterbox	Turbo	Late 2025	11K+	MIT	Zero-shot voice cloning from 5s audio; emotion control; paralinguistic tags; sub-200ms on GPU
GPT-SoVITS	v2pro	Jun 6, 2025	55.7K	MIT	1-minute voice cloning; cross-lingual (EN/ZH/JA/KO); v4 models at 48kHz native
Piper TTS	v1.4.1	Feb 5, 2026	7K+	GPL-3.0	Fast ONNX neural TTS; 100+ voices; runs on anything; moved to OHF-Voice/piper1-gpl
					341M params;

Stable Audio Open	Small	Early 2026	3K+	Stability Community	11s audio in <8s on ARM; optimized for Apple Silicon via Arm KleidiAI
AudioCraft/MusicGen	v1.4.0a1	Maintenance	22K+	MIT/CC-BY-NC	⚠️ No major updates since 2024; works on MPS but not optimized
Bark	April 2023	Archived	36K+	MIT	⚠️ Effectively unmaintained; Suno pivoted to commercial

Actionable insight — ACE-Step 1.5: This is the music generation breakthrough of Q1 2026. Hybrid LM + Diffusion Transformer architecture GitHub generates full songs with vocals, instruments, and lyrics in **50+ languages**. Supports LoRA fine-tuning for style adaptation. Runs on Mac per README. Outperforms Suno v5 on the SongEval benchmark (8.09). NYU Shanghai RITS ComfyUI integration available. **This didn't exist 6 months ago.**

Actionable insight — mlx-audio v0.4.1: Supports **9 TTS models** including Kokoro, Qwen3-TTS, Sesame, Spark, Dia, Chatterbox, and Outetts — all through one unified API. PyPI GitHub The OpenAI-compatible server means any agent can call it like it's calling OpenAI's TTS endpoint. `pip install mlx-audio && mlx_audio.server` and you have a local TTS API.

Actionable insight — Chatterbox Turbo: The 350M distilled model with paralinguistic tags (`[laugh]` , `[cough]` , `[chuckle]`) makes characters feel alive. GitHub Zero-shot cloning from 5 seconds of reference audio. Resemble AI Now integrated into mlx-audio, so you get it through the same API.

Bottom Line for Timmy: Install **mlx-audio v0.4.1** (`pip install mlx-audio`) as your unified voice engine — it wraps Kokoro, Chatterbox, and others behind one OpenAI-compatible API. For music, clone and run **ACE-Step 1.5** (`git clone https://github.com/ace-step/ACE-Step-1.5`) — it generates full songs with vocals on your M3 Max in under 10 seconds. NYU Shanghai RITS For character voices, use **Chatterbox Turbo** via mlx-audio with a 5-second reference clip per character. For bulk narration, **Piper v1.4.1** (`pip install piper-tts`) is the fastest lightweight option PyPI at the cost of naturalness.

---

5. Agent orchestration converged on MCP as the universal protocol

FastMCP **powers 70% of MCP servers**. PyPI Every major framework now speaks MCP natively. The practical question is no longer "which framework" but "how many MCP servers do you need."

Tool	Version	Released	Stars	License	What it does
FastMCP	3.1.1	Mar 14, 2026	~10K	Apache-2.0	Standard MCP server/client framework; 3.1 added CodeMode (meta-tools with BM25 search)
CrewAI	1.11.0	Mar 18, 2026	45.9K	MIT	Role-based agent orchestration; Crews + Flows paradigms; native MCP via MCPServerAdapter
Agno	v2.5.10	Mar 17, 2026	38.8K	Apache-2.0	Fast multi-modal runtime; AgentOS backend; 100+ tools; MCPTools with parallel calls
PocketFlow	~0.1.x	Active	~30K	MIT	100-line Graph + Shared Store framework; zero dependencies; ported to 7 languages
LangGraph	1.0 LTS	Ongoing	25K	MIT	Graph-based stateful orchestration; Deep Agents v0.4 for research workflows
OpenAI Agents SDK	Active	Mar 2025+	19K+	MIT	Lightweight Swarm successor; 5 primitives; provider-agnostic via LiteLLM
Smolagents	v1.18.0	Mar 2026	~15K	Apache-2.0	HuggingFace's code-first agent library; ~1K lines core; sandboxed execution

Google ADK	2.0 Alpha	Feb 2026	Growing	Apache-2.0	Code-first framework; AgentTeam API; 30+ third-party integrations

Actionable insight — FastMCP 3.1.1: The **CodeMode** feature is new and significant — it exposes meta-tools that let LLMs write multi-step tool scripts with BM25 search over your tool catalog. GitHub This means an agent can discover and compose tools dynamically rather than having every tool pre-loaded in context. Also: **granular async authorization**, OpenTelemetry tracing, and background tasks with Redis. FastMCP `pip install fastmcp==3.1.1`.

Actionable insight — PocketFlow: At 100 lines with zero dependencies, this is the framework you should actually understand. GitHub The Graph + Shared Store pattern is the minimal viable agent architecture. Pocket Flow GitHub Fork it, read all 100 lines, then use it as the skeleton for Timmy's core orchestration. Everything else (CrewAI, Agno) adds convenience at the cost of opacity.

Actionable insight — CrewAI 1.11.0: The **MCPServerAdapter** wraps any MCP server as a CrewAI tool. PyPI The Flows paradigm (event-driven, deterministic) complements Crews (autonomous). GitHub For Timmy Time, Flows handle the predictable pipeline (image → music → video → publish) while a Crew handles the creative decisions.

Bottom Line for Timmy: Build your MCP servers with **FastMCP v3.1.1** (`pip install fastmcp==3.1.1`) — one server each for: image generation (wrapping ComfyUI API), voice synthesis (wrapping mlx-audio), music generation (wrapping ACE-Step), Nostr publishing (wrapping nostr-sdk), and Lightning payments (wrapping LND). Use **PocketFlow** as the orchestration skeleton — read and understand all 100 lines, then extend. If you need role-based multi-agent behavior, layer **CrewAI 1.11.0** on top with MCPServerAdapter connecting to your FastMCP servers.

---

6. The Nostr developer stack reached minimum viability

nostr-sdk Python bindings handle signing, relay management, NWC, and encrypted messaging. nostrdvm provides the NIP-90 marketplace scaffold. Blossom handles media storage. The full agent social layer is buildable today.

Tool	Version	Released	Stars	License	What it does
nostr-sdk	v0.44.2	Jan 29,	2K+	MIT	Rust-backed Python bindings; relay pool, NWC client,

(Python)		2026			embedded Tor, NIP support matrix
nostrdvm	Active dev	Ongoing	~53	Open source	NIP-90 Data Vending Machine framework; LNbits wallet integration; auto key generation
Blossom	Spec stable	Adopted	Moderate	Public Domain	HTTP blob storage via SHA-256 hash; Nostr key auth; natively in Primal, Amethyst, Nostur
Nostrify	Active	2025–2026	Moderate	Open source	Comprehensive toolkit: Blossom uploader, relay management, event handling
Alby MCP	New	Q1 2026	New	MIT	Connect Lightning wallet to any MCP-compatible LLM via NWC
startwithbitcoin	New	Q1 2026	New	Open source	Framework giving AI agents Nostr identity + NWC wallet + Lightning payments

Actionable insight — nostr-sdk v0.44.2: `pip install nostr-sdk` PyPI gives you pre-built wheels for macOS. Rust The built-in NWC client means you don't need a separate NIP-47 library — wallet operations (pay invoice, make invoice, get balance) are part of the SDK. This is the **only Python library you need** for Nostr + NWC.

Actionable insight — nostrdvm: This is Timmy's economic primitive. A NIP-90 DVM lets any Nostr user request compute (image generation, music creation, transcription) and pay via Lightning. GitHub Timmy publishes capabilities, receives jobs, executes them, and gets paid — all over Nostr relays. The framework handles the protocol; you supply the compute functions. GitHub

Actionable insight — startwithbitcoin: This new framework is literally "give an AI agent identity (Nostr keypair), a wallet (NWC), payments (Lightning), and communication

(encrypted DMs)" in one package. GitHub Has a Claude Code skill. **This is the closest thing to a Timmy Time starter kit.**

Bottom Line for Timmy: `pip install nostr-sdk==0.44.2`. Generate a keypair, publish a kind:0 profile event, and set up NWC connection to your LND node. Clone **nostrdvm** and register Timmy as a NIP-90 Data Vending Machine GitHub offering image generation (kind:5100) and text generation (kind:5050). Use **Blossom** via nostr-sdk to upload generated media to `blossom.nostr.build` (free up to 20MB). Nostr Timmy now has a Nostr identity, can receive jobs, execute them, get paid over Lightning, and post results — all sovereign, all self-hosted.

---

7. Lightning agent-tools made machine payments a solved problem

Lightning Labs open-sourced a complete agent toolkit in February 2026. The **L402 protocol** (HTTP 402 + Lightning invoice + macaroon) is now the standard for AI agent micropayments.

Tool	Version	Released	Stars	License	What it does
LND	v0.20.1-beta	Feb 10, 2026	7.8K+	MIT	Full Lightning node; channel graph migrated to SQL; DNS address support
Lightning agent-tools	v1	Feb 11, 2026	New	MIT	7 composable skills + MCP server; **lnget** for L402-aware HTTP; aperture for paid endpoints
Alby SDK	v7.0.0	Q4 2025	577	MIT	JS/TS SDK for NWC + Alby API; 2-line Lightning payments
Alby Hub	Active	Q1 2026	Active	MIT	Self-custodial LDK-based Lightning node with NWC
LNbits	v1.4 "DNI"	Q1 2026	2.5K+	MIT	Lightweight wallet server; 137+ extensions; Boltz-powered standalone Docker

BTCPay Server	v2.3.x	Q1 2026	7.4K	MIT	Self-hosted payment processor; 30+ e-commerce plugins; Lightning native
Cashu/nutshell	v0.17.0	Active	470	MIT	Chaumian ecash wallet/mint; NUT-21/22 auth; Python reference implementation

Actionable insight — Lightning agent-tools: The **lightning-mcp-server** exposes 18 read-only LND tools via MCP. **lnget** is `wget` with automatic L402 payment — your agent can access any L402-gated API by just calling `lnget <url>`. **Aperture** is the reverse proxy for hosting your own paid endpoints. Together, Timmy can both consume and sell AI services for sats. **This stack didn't exist 6 weeks ago.**

Actionable insight — LNbits v1.4: The Boltz-powered standalone Docker image GitHub means you can spin up a **non-custodial LNbits** instance with zero external dependencies. UV dependency management makes installation 10x faster. The extension ecosystem (137+ repos) gives you LNURL-pay, NFC cards, point-of-sale, and more. Active bounties for NWC integration.

Actionable insight — Cashu v0.17.0: Ecash tokens are the **privacy-preserving payment layer** for agent-to-agent transactions. `pip install cashu` gives you a mint and wallet. NUT-19/20/21/22 add authentication and cached responses. For Timmy Time, Cashu tokens can be the internal currency between agents without touching the blockchain for every transaction.

Bottom Line for Timmy: Run **LND v0.20.1-beta** as your Lightning node. Install **lightning-agent-tools** and start the MCP server for agent access to node operations. Set up **Aperture** as a reverse proxy in front of Timmy's API endpoints to gate them with L402 — anyone (human or AI) can pay-per-request. Run **LNbits v1.4** (`docker run lnbits/lnbits`) connected to LND for extension-based wallet management. Install **Cashu nutshell v0.17.0** (`pip install cashu`) for privacy-preserving inter-agent micropayments.

---

8. Knowledge graphs and memory became agent-native

Graphiti hit **94.8% on the DMR benchmark**, arXiv making temporal knowledge graphs the state-of-the-art for agent memory. Mem0 reached v1.0 with 50+ provider integrations. Socket DeepWiki ChromaDB ships weekly. PyPI

Tool	Version	Released	Stars	License	What it does
Graphiti (Zep)	v0.28.2	Mar 2026	23.3K	Apache-2.0	Bi-temporal knowledge graph; sub-second retrieval; hybrid semantic + BM25 + graph traversal
Mem0	v1.0.5	Mar 3, 2026	50.6K	Apache-2.0	Self-improving memory layer; +26% accuracy over OpenAI Memory; 22+ vector stores
Neo4j	2026.02	Feb 2026	N/A	GPL-3/Commercial	GraphRAG v1.13.0; SimpleKGPipeline; ToolsRetriever for agent tool selection
ChromaDB	v1.5.5	Mar 10, 2026	~17K	Apache-2.0	AI-native embedding database; weekly releases; bloom filter optimizations
Qdrant	v1.16.x	Mar 2026	~23K	Apache-2.0	Rust vector search; 24x compression via asymmetric quantization; Qdrant Edge for in-process
LangChain	1.0 LTS	Oct 2025	97K	MIT	`langchain-neo4j` v0.8.0 for deep graph integration; runs on LangGraph runtime

Actionable insight — Graphiti v0.28.2: The bi-temporal model tracks both **when an event happened** and **when Timmy learned about it**. Emergent Mind Facts have validity windows — old facts are invalidated but preserved for reasoning about what changed. GitHub Supports Neo4j, FalkorDB, and Kuzu as backends. GitHub Has an **MCP server** for Claude/Cursor integration. For a creative AI agent that needs to remember character relationships, story arcs, and user preferences over time, this is purpose-built.

Actionable insight — Mem0 v1.0: Reached the v1.0 milestone. The memory pipeline (extract facts → similarity search → conflict resolution → ADD/UPDATE/DELETE) runs fully self-hosted with Qdrant as the default vector store and Neo4j for graph memory. DeepWiki

91% faster and **90% fewer tokens** than naive context stuffing. MCP server available (`mem0-mcp`).

Actionable insight — Qdrant Edge: The in-process version runs embedded in your Python application — no separate server needed. Compatible with Qdrant Cloud snapshots, so you can develop locally and deploy to cloud if needed. For a sovereign setup, this eliminates one more external dependency.

Bottom Line for Timmy: Deploy **Graphiti v0.28.2** (`pip install graphiti-core`) backed by Neo4j Community Edition for Timmy's long-term memory — character relationships, user preferences, story continuity. Use **Mem0 v1.0.5** (`pip install mem0ai`) with Qdrant as the vector store for session-level conversational memory. Run **ChromaDB v1.5.5** (`pip install chromadb`) for document retrieval/RAG over reference materials. The three-layer stack: ChromaDB for documents, Mem0 for conversations, Graphiti for temporal knowledge graph.

---

9. Video and streaming tools are ready for AI-driven production

LiveKit Agents SDK v1.4.3 supports voice-to-voice AI pipelines natively. MediaMTX handles every streaming protocol. The missing piece — AI video generation locally — is filled by Wan 2.2 and LTX-2 in ComfyUI.

Tool	Version	Released	Stars	License	What it does
LiveKit	v1.9.12 / Agents v1.4.3	Mar 5, 2026	High	Apache-2.0	WebRTC SFU with built-in agent framework; STT→LLM→TTS voice pipelines
MediaMTX	v1.16.3	Mar 1, 2026	18.1K	MIT	Zero-dependency media server; SRT, WebRTC, RTSP, RTMP, LL-HLS
OBS Studio	v32.0.4	Mar 2026	62K+	GPL-2.0	obs-websocket v5.x built-in; partial Canvas API in 32.1.0 RC
MoviePy	v2.1.2	Stable	12K+	MIT	Python video editing; numpy-based compositing
Wan 2.2	Active	Q1 2026	N/A	Open	MoE text-to-video/image-to-video in ComfyUI and Draw Things

LTX-Video 2	Active	Q1 2026	N/A	Open	Fast unified audio+video generation; 12-16GB VRAM

Actionable insight — LiveKit Agents v1.4.3: Drop Python 3.9 (requires 3.10+), adds `AgentConfigUpdate` tracking, browser navigation via RPC, and plugins for Speechmatics, Neuphonic, and AssemblyAI. Supports direct speech-to-speech models. For Timmy's live streaming presence, this is the real-time interaction layer.

Actionable insight — MediaMTX v1.16.3: The new "always available streams" feature shows an offline video placeholder while the publisher is absent. Combined with self-upgrader (`./mediamtx --upgrade`), this is a zero-maintenance streaming relay for Timmy's persistent broadcast presence.

Bottom Line for Timmy: Run **MediaMTX v1.16.3** as your streaming relay (supports every protocol). Control **OBS Studio** via obs-websocket v5.x for scene composition — Python library `obsws-python` lets agents switch scenes, manage sources, trigger transitions. Use **LiveKit Agents SDK v1.4.3** for real-time voice interaction with viewers. Generate video clips with **Wan 2.2** via ComfyUI, edit with **MoviePy**, and stream through MediaMTX. The full pipeline: ComfyUI generates → MoviePy composites → MediaMTX streams → OBS mixes → LiveKit handles audience interaction.

---

10. Self-hosting infrastructure now rivals cloud developer experience

Coolify v4 deploys 280+ services with one click. n8n 2.0 has MCP integration. Tailscale Aperture secures AI agents with identity-aware networking. The sovereign PaaS stack is production-grade.

Tool	Version	Released	Stars	License	What it does
n8n	v2.12.2	Mar 13, 2026	50K+	Fair-code	AI workflow automation; 400+ integrations; MCP webhooks; Draft/Publish system
Coolify	v4.0.0-beta.462	Jan 2026	35K+	Apache-2.0	Self-hosted PaaS; 280+ one-click services; Nixpacks deployment
Dokku	v0.37.7	Q1 2026	29K+	MIT	Smallest PaaS; Docker + git-push; Railpack builder plugin

Tailscale	v1.96.2	Mar 2026	20K+	BSD-3-Clause	Mesh VPN; **Aperture** (alpha) AI gateway for securing LLM sessions; Peer Relays GA
Forgejo	v14.0.3	Mar 9, 2026	N/A	MIT	Self-hosted forge; GitHub Actions-compatible CI/CD

Actionable insight — Tailscale Aperture: This alpha feature is purpose-built for Timmy Time's architecture. It secures and monitors AI agent LLM sessions using Tailscale identity — no more API key management across services. Every service on your tailnet gets identity-aware access to every other service without managing credentials.

Actionable insight — n8n v2.12.2: MCP webhook integration means n8n can be triggered by any MCP-compatible agent. The Draft/Publish system prevents accidental live-editing disasters. Task Runners provide sandboxed Python/JS execution. For Timmy's workflow automation (schedule posts, process payments, trigger content generation), n8n is the visual orchestration layer that non-developers can modify.

Bottom Line for Timmy: Install **Tailscale** on every device in your stack and enable **Aperture** for AI gateway security. Deploy **Coolify v4** (`curl -fsSL https://cdn.coolify.io/install.sh | bash`) on a dedicated server as your PaaS for managing services. Run **n8n v2.12.2** (`docker run -it --rm n8nio/n8n`) for visual workflow automation with MCP webhook triggers. Host code on **Forgejo v14.0.3**. All services communicate over your Tailscale mesh — zero exposure to the public internet except intentional endpoints.

---

11. Game engines gained LLM agent integration paths

Godot RL Agents bridges reinforcement learning training with Godot 4.x. GamingAgent (ICLR 2026) provides standardized LLM↔game interfaces. Luanti's Lua API is architecturally ready for AI NPCs.

Tool	Version	Released	Stars	License	What it does
Godot	v4.6.1	Feb 16, 2026	100K+	MIT	Open-source game engine; v4.6 added LibGodot (engine as library), Jolt physics default
Godot RL	Active	Ongoing	Active	MIT	Python↔Godot bridge for deep RL;

Agents					StableBaselines3, Ray RLLib; ONNX export
Luanti	v5.15.1	Feb 8, 2026	12.4K	LGPL-2.1+	Voxel game platform; Lua scripting; dynamic shadows in v5.15
OpenMW	v0.50.0	Late 2025	~6K	GPL-3.0	Morrowind reimplementation; extensive Lua 5.1 API; hot-reload; sandboxed execution
GamingAgent	Active	ICLR 2026	New	Open	LLM/VLM agents in standardized game environments; computer-use testing
LLM for Unity	Active	2025–2026	Moderate	Open	llama.cpp-based LLM integration in Unity; RAG for NPC knowledge

Actionable insight — Godot 4.6 + LibGodot: LibGodot lets you **embed the Godot engine as a library** in external applications. This means Timmy Time can programmatically drive a Godot game world from Python — spawn entities, control cameras, trigger animations — without the editor. Combined with Godot RL Agents for training and ONNX export for inference, you can train agent behaviors, export them, and run them in production without Python.

Actionable insight — Luanti v5.15.1: The server-client Lua architecture (global scripts on server, local scripts per entity) maps naturally to AI agent integration. Timmy could host a Luanti server where AI agents control NPCs via the Lua API while humans play alongside them. No dedicated AI framework needed — the scripting layer is sufficient.

Bottom Line for Timmy: Install **Godot 4.6** and the **Godot RL Agents** plugin (`pip install godot-rl`) for any virtual world Timmy inhabits. Use LibGodot to embed the engine in your Python agent pipeline. For a voxel sandbox world, run **Luanti v5.15.1** and write Lua scripts that bridge to your agent's decision-making via HTTP/socket calls to your MCP servers. For immersive RPG content, **OpenMW v0.50.0**'s Lua API with hot-reload lets you iterate on NPC behavior without restarting.

---

12. Decentralized identity for AI agents is crystallizing around

three specs

did:nostr provides DID resolution from Nostr keypairs. SoulSpec v0.4 defines agent personas in portable Markdown. L402 adds economic identity through cryptographic proof of payment.

Tool	Version	Status	License	What it does
did:nostr	Draft	W3C Community Group draft	Public	DID method resolving Nostr pubkeys; offline-first + relay-enhanced resolution
SoulSpec / SOUL.md	v0.4	Widely adopted	Open	Agent persona standard; Markdown + JSON; used by OpenClaw (145K stars), Claude Code, Cursor
lnurl (Python)	v0.8.3	Stable	MIT	LNURL encode/decode/auth; maintained by LNbits team
Soul Protocol	Active	New	Open	Verifiable identity registry for AI agents; `did:soul:` DIDs; Ed25519; REST API
L402	bLIP-0026	Formalized	Open	HTTP 402 + Lightning payment as cryptographic identity; transaction-based auth
startwithbitcoin	New	Q1 2026	Open	Nostr keypair + NWC wallet + Lightning payments for agent identity/agency

Actionable insight — SoulSpec v0.4: This is the de facto standard. Create `soul.json` (manifest), `SOUL.md` (personality/values), `IDENTITY.md` (public identity), and `AGENTS.md` (workflows). Compatible with OpenClaw, Cursor, Windsurf, Claude Code, and custom pipelines. MSR 2026 academic paper validates the approach across 466 open-source projects. **SoulGen** (soulgen.dev) auto-generates these files.

Actionable insight — did:nostr: The spec supports three resolution strategies — offline-first (generate DID document from pubkey alone, no network needed), HTTP via `.well-known` , and relay-enhanced (query Nostr relays for metadata). For Timmy, `did:nostr:<hex-pubkey>` is the decentralized identifier that works even when relays are down.

Actionable insight — L402 as identity: Lightning Labs is positioning L402 as **transaction-based identity** — cryptographic proof of payment replaces API keys and accounts. An agent proves its identity by paying, not by authenticating. This aligns perfectly with sovereign AI agents that shouldn't depend on centralized identity providers.

Bottom Line for Timmy: Create Timmy's identity stack today. Write a **SOUL.md** following SoulSpec v0.4 (use soulgen.dev as a starting point). Generate a Nostr keypair and derive `did:nostr:<pubkey>`. Publish kind:0 profile and kind:10002 relay list events. Install `pip install lnurl==0.8.3` for LNURL-auth on any endpoints Timmy exposes. The identity triple is: **did:nostr** (who Timmy is), **SoulSpec** (what Timmy believes), **L402** (how Timmy transacts). All three are decentralized, self-hosted, and operational today.

Conclusion: the sovereign creative AI agent stack is deployable now

Six months ago, building a self-hosted creative AI agent required stitching together half-broken prototypes. Today, every layer has **at least one production-quality, actively maintained, MIT/Apache-licensed tool** that runs on an M3 Max. The three biggest shifts in Q1 2026: **MLX became definitively faster than llama.cpp** for Apple Silicon inference, **Lightning Labs open-sourced agent-tools** making machine-to-machine payments trivial, and **ACE-Step 1.5 cracked open-source music generation** that rivals commercial services.

The critical path for Timmy Time this month is not more research — it's integration. The stack: **vllm-mlx** for inference, **FastMCP** for tool exposure, **nostr-sdk** for social identity, **LND + lightning-agent-tools** for payments, **Graphiti** for memory, **mlx-audio** for voice, **ComfyUI** for images, **ACE-Step** for music, and **Tailscale** to tie it all together in a zero-trust mesh. Every component listed here has commits within 90 days, working releases, and runs on Apple Silicon today. The window between "possible" and "obvious" is closing fast.