DAWZY: A New Addition to AI powered "Human in the Loop" Music Co-creation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Digital Audio Workstations (DAWs) offer fine control, but mapping high-level intent (e.g., "warm the vocals") to low-level edits breaks creative flow. Existing artificial intelligence (AI) music generators are typically one-shot, limiting opportunities for iterative development and human contribution. We present *DAWZY*, an open-source assistant that turns natural-language (text/voice/hum) requests into reversible actions in REAPER. *DAWZY* keeps the DAW as the creative hub with a minimal GUI and voice-first interface. *DAWZY* uses LLM-based code generation as a novel way to significantly reduce the time users spend familiarizing themselves with large interfaces, replacing hundreds of buttons and drop downs with a chat box. DAWZY also uses three Model Context Protocol tools for live state queries, parameter adjustment, and AI beat generation. It maintains grounding by refreshing state before mutation; and ensures safety and reversibility with atomic scripts and undo. In evaluations, DAWZY performed reliably on common production tasks and was rated positively by users across Usability, Control, Learning, Collaboration, and Enjoyment. We show reliability on common production tasks; code and a short demo are available. [1]

1 Introduction

Modern music production centers on Digital Audio Workstations (DAWs) Leider [2004], which democratize pro-quality creation but burden users with option overload that disrupts flow [Kjus, 2024]. A gap persists between high-level intent (e.g., "make the vocals warmer") and the low-level steps to realize it.

Prior work points to a path forward: mature DAW scripting (Ableton's Max for Live/Live API [Ableton, 2024], REAPER's ReaScript/JSFX [Cockos Incorporated, 2024]), co-creative agents inside production loops (e.g., Juice [Bricard et al., 2024]), fully generative systems largely outside fine-grained editing (e.g., Suno [Suno, 2024]), advances in code generation [Chen et al., 2021], and standardized tool invocation via the Model Context Protocol (MCP) [Anthropic, 2024, Hou et al., 2025].

We introduce *DAWZY*, an open-source assistant that maps natural-language requests to precise, context-aware, reversible ReaScript actions in REAPER. DAWZY queries live session state, emits auditable edits, and favors a minimal-GUI, voice-first workflow with plain-language explanations to support learning. It primarily interacts with REAPER through LLM code generation. Related efforts (e.g., Mozart AI [Mozart AI, 2025]) explore closed-source adjacent ideas; DAWZY emphasizes open-source availability and ReaScript-specific reliability, complementing rather than replacing existing tools.

Primary Contributions

[1] **Resources:** Code (anonymous) Demo (anonymous)

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

- **System design & open-source prototype.** REAPER-targeted pipeline mapping natural language to safe, reversible ReaScript grounded in live state (Sec. 2).
- **MCP tool suite.** Permissioned tools for state query, unit-consistent FX parameter adjustment, and AI beat generation; supports future cross-DAW portability (Sec. 2.2).
- **Minimal-GUI, voice-first interaction.** Natural-language control with buttons for common tasks("start," "stop," "record," "undo") to reduce GUI micromanagement (Sec. 2.1).
- **Explain-as-you-go pedagogy.** Plain-language rationales accompany each edit to support learning and auditability.
- **ReaScript-focused model adaptation.** Plan to fine-tune an open-source LLM for reliable REAPER code generation (Sec. 4).

2 DAWZY Architecture

DAWZY comprises three layers (Figure 1): *User Interaction*, *Processing*, and *Execution*, which capture natural-language intent, interpret it, and translate it into precise DAW operations.

2.1 User Interaction Layer

The User Interaction Layer is a minimal-GUI entry point for expressing intent via **text**, **speech**, or **humming**, mediated by an **Electron.js** app [OpenJS Foundation, 2024]. Given the complexity of traditional DAW interfaces, DAWZY prioritizes direct, natural-language control to reduce GUI micromanagement. (1) **Text** — Users type commands/questions in Electron; queries are forwarded as text. (2) **Speech** - Spoken commands are transcribed by **Whisper** [Radford et al., 2022] and follow the same downstream path (hands-free). (3) **Humming** - A "record hum" button captures sketches; a local **BasicPitch** pipeline [Spotify] converts audio to MIDI, which is auto-imported into REAPER as a new track.

2.2 Processing Layer

The Processing Layer turns user input into context-aware DAW operations. Off-the-shelf LLM approaches often hallucinate commands, mis-index tracks/parameters, or ignore live state. DAWZY constrains behavior via a reliable LLM, and context grounding.

- **Electron gateway.** Routes all queries to the LLM and returns responses/confirmations; hummed audio is sent to the hum-to-MIDI pipeline.
- **LLM.** We use **OpenAI GPT-5** [OpenAI, 2025a,b] to interpret intent, call MCP tools, and emit Lua ReaScript. Open-source baselines (e.g., Qwen3-Coder-480B-A35B-Instruct [Qwen Team, Team, 2025]) underperformed, frequently producing invalid indices when the full context (track/parameter mappings) was not considered; GPT-5 generated reliable edits.
- **Model Context Protocol (MCP).** Exposes DAW capabilities as explicit, permissioned functions between the LLM and REAPER:
  - **State query.** Enumerates tracks, items, FX, and routing to ground edits in live session state and keep tool calls synchronized.
  - **FX parameterization (`fxparam`).** Converts human units (dB, ms) to ReaScript slider ranges (e.g., 0–1, 0–4) to prevent scaling errors. Code generation failed here because the LLM could not reliably convert between units.
  - **Beat generation.** Meta's **MusicGen-small (300M)** model is run locally to create an audio waveform based on a text description [Meta AI, Copet et al., 2023].
- **Hum to MIDI.** The open-source **Spotify BasicPitch** model is run locally to convert hums into MIDI data [Spotify, Bittner et al., 2022].

2.3 Execution Layer

The Execution Layer (Figure 1) performs edits in REAPER safely, reversibly, and transparently by grounding actions in live project state. (1) **ReaScript actuation** - GPT-5 generates Lua that ReaPy

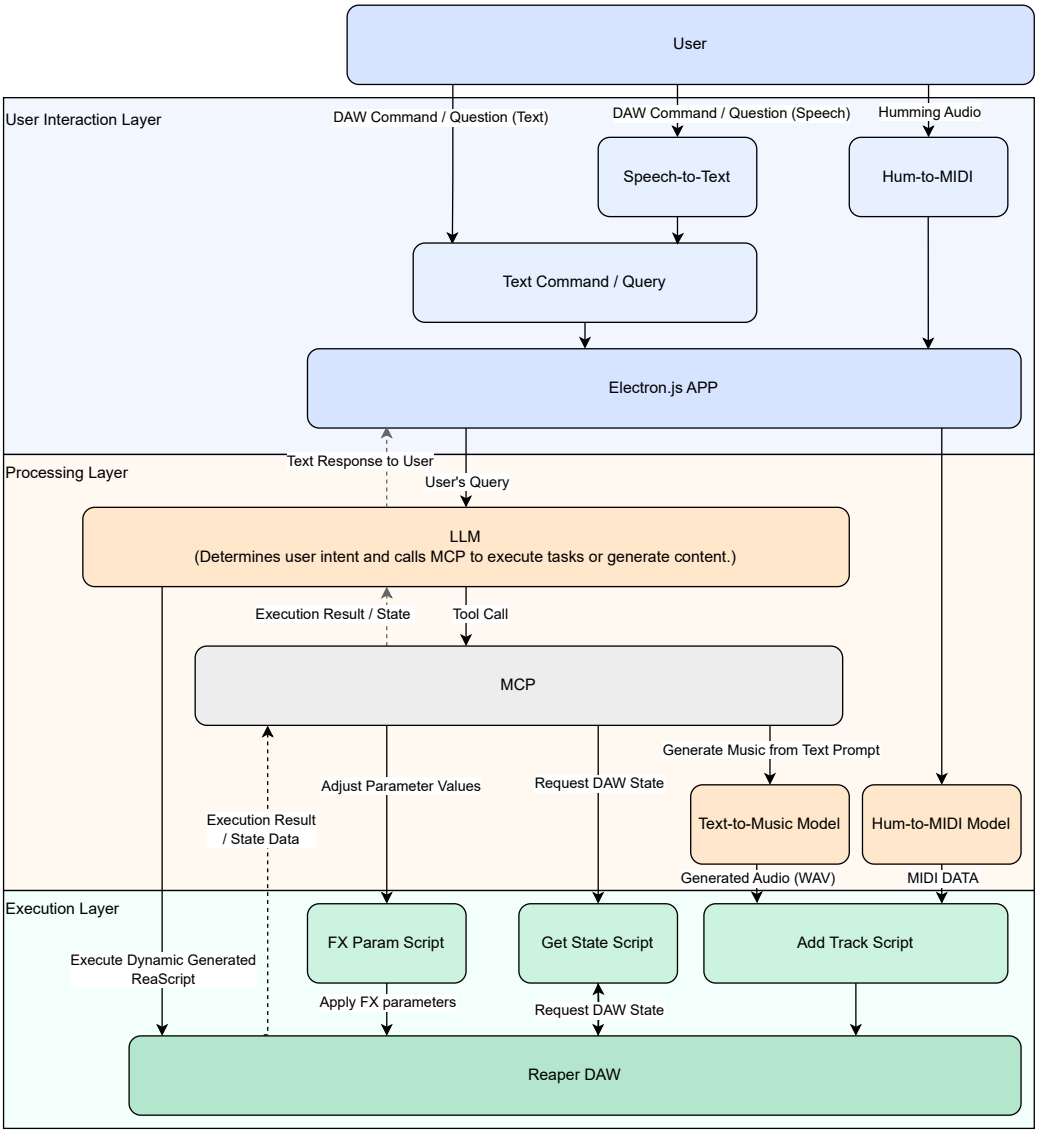

Figure 1: **DAWZY Architecture.** User intent (text/speech/hum) flows through the Electron gateway to the LLM and MCP tools, then executes as reversible ReaScripts in REAPER. Rounded rectangles denote AI/MCP components; sharp rectangles denote DAW/runtime components; dashed arrows indicate data queries; solid arrows indicate state-changing actions.

executes to modify the project; changes are reversible. (2) **Utility scripts** - Specialized scripts handle (i) FX parameter updates, (ii) project-state summaries, and (iii) audio/MIDI import as new tracks.

3 Evaluation

We evaluate DAWZY using both objective performance tasks and subjective user ratings.

3.1 Objective Evaluation

To test reliability, we designed four reproducible tasks: (1) **Multi-instruction FX processing** — "Double the first track's volume, increase the decay, and set the attack to 10 ms," (2) **GUI navigation** — "Open the FX browser for the first track," (3) **Workflow automation** — "Duplicate the first track, pitch it up one octave, and blend it in at 20%," and (4) **Educational interaction** — "What does attack time do in the second track's compressor?"

3.2 Model Comparison

Building on the four tasks described in Sec. 3 (Objective Evaluation), we ran 3 trials per task for each model. Open-source baselines (QWEN-480B, GPT-OSS-120B, GPT-OSS-20B) achieved only 25–50% success, often failing due to hallucinated or invalid ReaScript functions and mis-indexed parameters. GPT-5, by contrast, reached 83% success, benefiting from broader ReaScript coverage and stronger reasoning, which motivated our shift toward it for reliable integration.

Task	QWEN-480B	GPT-OSS-120B	GPT-OSS-20B	GPT-5
FX (1)	2	2	0	2
GUI (2)	1	0	0	2
Chain (3)	0	0	0	3
Learn (4)	3	3	3	3
Success Rate	50%	42%	25%	83%

Table 1: **Objective task success across models.** Scores denote successful trials out of three per task; "Success Rate" is over all four tasks.

3.3 Subjective Evaluation (MOS Test)

We conducted a **Mean Opinion Score (MOS)** test with *21* participants, who rated DAWZY on a 5-point Likert scale across five dimensions: **Usability**, **Control**, **Learning**, **Collaboration**, and **Enjoyment**. All dimensions scored above the neutral threshold of 3, with *Enjoyment* ($M = 4.48$) and *Learning* ($M = 4.38$) rated highest, followed by *Collaboration* ($M = 4.29$), *Usability* ($M = 4.14$), and *Control* ($M = 3.81$), reflecting a positive overall perception of the system.

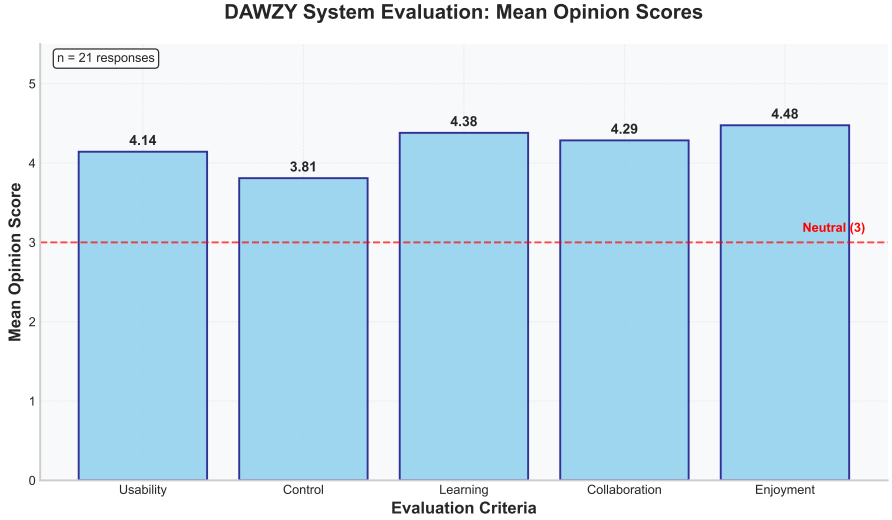

Figure 2: Mean Opinion Score (MOS) results for DAWZY (N=21). The dashed red line indicates the neutral rating (3).

4 Conclusion

DAWZY demonstrates natural-language control of complex creative software can augment—rather than replace—human creativity. Pairing state extraction with context-aware code generation, it performs precise, reversible ReaScript edits inside REAPER while keeping users in the loop. Current constraints arise from REAPER-specific APIs and the need to adapt the state-extraction layer per DAW; the prototype emphasizes core operations over advanced plugin interactions and routing. As DAWs expand scripting and APIs [Ableton AG, 2025] and code generation improves [Alenezi and Akour, 2025, p. 2], we anticipate broader integration in professional workflows.

A key contribution is treating AI not as a black box but as a transparent, editable component. Looking ahead, we will (i) conduct user studies; (ii) expand plugin/routing support and cross-DAW portability via abstraction layers; and (iii) train and fine-tune open-source models explicitly for reliable ReaScript synthesis, supported by a curated (intent, state, script) corpus, unit tests/undo-safety checks, and public benchmarks against closed models. We invite the community to try the demo, provide feedback, and collaborate on this open framework for natural-language creative tool control.

References

Colby N. Leider. *Digital Audio Workstation*. McGraw-Hill, Inc., USA, 1 edition, 2004. ISBN 0071422862.

Yngvar Kjus. The platformization of music production: How digital audio workstations are turned into platforms of labor market relations. *New Media & Society*, 2024. doi: 10.1177/14614448241304660. First published online December 11, 2024.

Ableton. Ableton reference manual, version 12 — max for live, 2024. URL https://www.ableton.com/en/manual/max-for-live/. Accessed 2025-08-18.

Cockos Incorporated. Reascript documentation. https://www.reaper.fm/sdk/reascript/reascript.php, 2024. Accessed 18 Aug 2025.

S. Bricard, F. D'Errico, M. Devis, A. Bitton, D. ADC, and E. Vincent. Juice: A framework for co-creative agents in digital audio workstations. arXiv:2402.19323, 2024. URL https://arxiv.org/abs/2402.19323.

Suno. Suno. https://suno.com/, 2024. Accessed 18 Aug 2025.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, and et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. URL https://arxiv.org/abs/2107.03374.

Anthropic. Introducing the model context protocol. https://www.anthropic.com/news/model-context-protocol, 2024. Accessed 18 Aug 2025.

Xinyi Hou, Yanjie Zhao, Shenao Wang, and Haoyu Wang. Model context protocol (mcp): Landscape, security threats, and future research directions, 2025. URL https://arxiv.org/abs/2503.23278.

Mozart AI. Mozart ai — ai-powered music production daw, 2025. URL https://getmozart.ai/. Product site; Accessed 2025-08-18.

OpenJS Foundation. Electron. https://www.electronjs.org/, 2024. Accessed 18 Aug 2025.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. *arXiv preprint arXiv:2212.04356*, 2022. doi: 10.48550/arXiv.2212.04356. URL https://arxiv.org/abs/2212.04356.

Spotify. Basic pitch — about. https://basicpitch.spotify.com/about. Product page; accessed 18 Aug 2025.

OpenAI. Introducing gpt-5. https://openai.com/index/introducing-gpt-5/, 2025a. Accessed 18 Aug 2025.

OpenAI. Gpt-5 system card. https://openai.com/index/gpt-5-system-card/, 2025b. Accessed 18 Aug 2025.

Qwen Team. Qwen3 models — hugging face collection. https://huggingface.co/collections/Qwen/qwen3-67dd247413f0e2e4f653967f. Collection page; accessed 18 Aug 2025.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Meta AI. facebook/musicgen-small. https://huggingface.co/facebook/musicgen-small. Hugging Face model card; accessed 18 Aug 2025.

Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez. Simple and controllable music generation, 2023.

Rachel Bittner, Min Kim, Juan José Bosch, Aren Jansen, Gordon Wichern, and Longshaop Hantrakul. A lightweight polyphonic pitch transcription model. *arXiv preprint arXiv:2203.09893*, 2022. URL https://arxiv.org/abs/2203.09893.

Ableton AG. Ableton live 12. https://www.ableton.com/, 2025. Accessed 18 Aug 2025.

Mamdouh Alenezi and Mohammed Akour. Ai-driven innovations in software engineering: A review of current practices and future directions. *Applied Sciences*, 15(3):1344, 2025. doi: 10.3390/app15031344. URL https://www.mdpi.com/2076-3417/15/3/1344.