

---

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

---

**Anonymous Author(s)**

Affiliation

Address

email

## Abstract

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

17 

## 1 Introduction

18     Modern music production centers on Digital Audio Workstations (DAWs) Leider [2004], which  
19     democratize pro-quality creation but burden users with option overload that disrupts flow [Kjus,  
20     2024]. A gap persists between high-level intent (e.g., “make the vocals warmer”) and the low-level  
21     steps to realize it.

22     Prior work points to a path forward: mature DAW scripting (Ableton’s Max for Live/Live API  
23     [Ableton, 2024], REAPER’s ReaScript/JFX [Cockos Incorporated, 2024]), co-creative agents inside  
24     production loops (e.g., Juice [Bricard et al., 2024]), fully generative systems largely outside fine-  
25     grained editing (e.g., Suno [Suno, 2024]), advances in code generation [Chen et al., 2021], and  
26     standardized tool invocation via the Model Context Protocol (MCP) [Anthropic, 2024, Hou et al.,  
27     2025].

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

35 

## Primary Contributions

---

<sup>1</sup>**Resources:** Code (anonymous) Demo (anonymous)

- **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).

## 46 2 DAWZY Architecture

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

### 49 2.1 User Interaction Layer

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

### 58 2.2 Processing Layer

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

- **Electron gateway.** Routes all queries to the LLM and returns responses/confirmations; hummed  
 63 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  
 65 ReaScript. Open-source baselines (e.g., Qwen3-Coder-480B-A35B-Instruct [Qwen Team, Team,  
 66 2025]) underperformed, frequently producing invalid indices when the full context (track/parameter  
 67 mappings) was not considered; GPT-5 generated reliable edits.
- **Model Context Protocol (MCP).** Exposes DAW capabilities as explicit, permissioned functions  
 69 between the LLM and REAPER:
  - **State query.** Enumerates tracks, items, FX, and routing to ground edits in live session state and  
 71 keep tool calls synchronized.
  - **FX parameterization (fxparam).** Converts human units (dB, ms) to ReaScript slider ranges  
 73 (e.g., 0–1, 0–4) to prevent scaling errors. Code generation failed here because the LLM could  
 74 not reliably convert between units.
  - **Beat generation.** Meta’s **MusicGen-small (300M)** model is run locally to create an audio  
 76 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  
 78 MIDI data [Spotify, Bittner et al., 2022].

### 79 2.3 Execution Layer

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

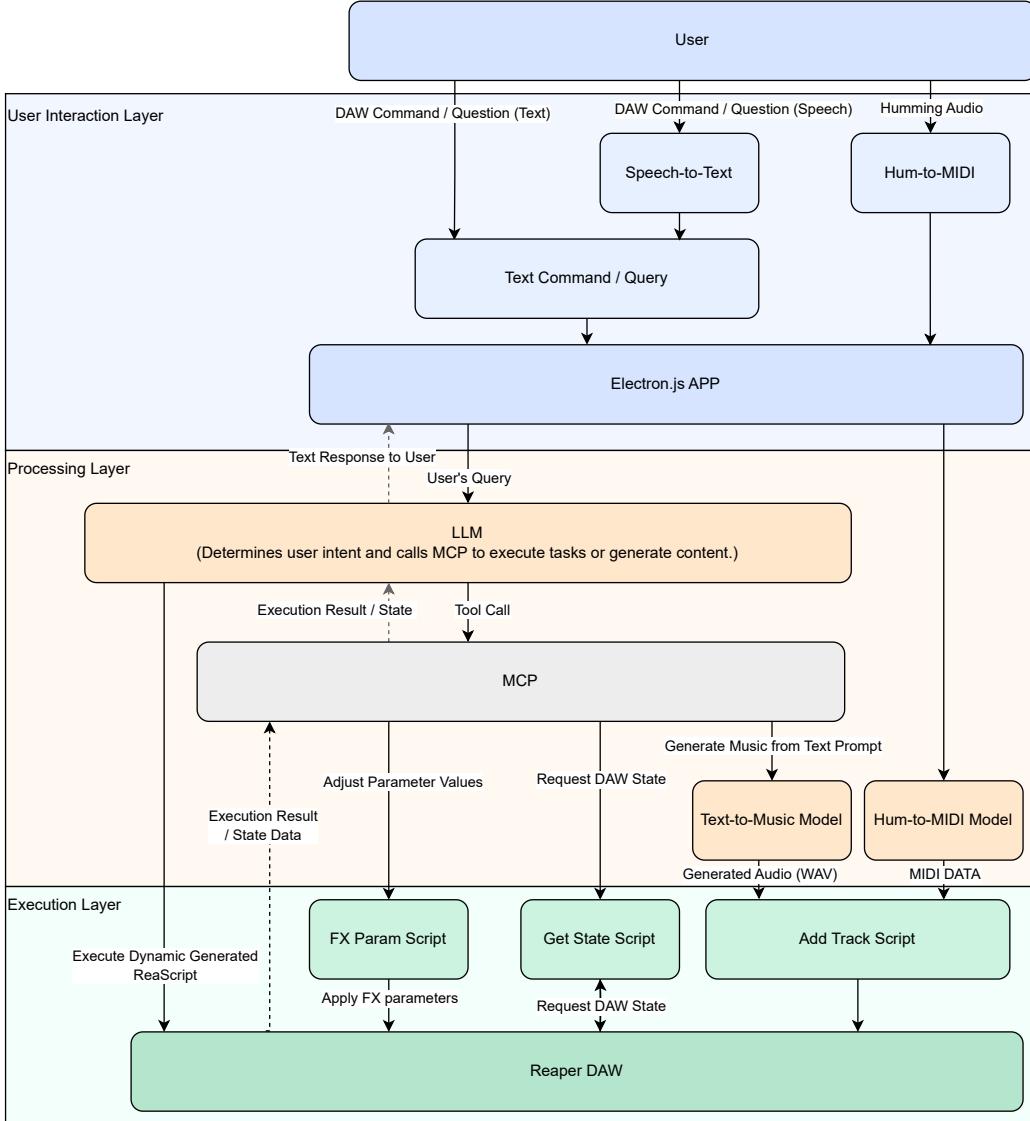


Figure 1: **DAWZY Architecture.** User intent (text/speech/hum) flows through the Electron gateway to the LLM and MCP tools, then executes as reversible ReaScripts in REAPER. Rounded rectangles denote AI/MCP components; sharp rectangles denote DAW/runtime components; dashed arrows indicate data queries;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

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

### 84 3 Evaluation

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

#### 86 3.1 Objective Evaluation

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

92 **3.2 Model Comparison**

93 Building on the four tasks described in Sec. 3 (Objective Evaluation), we ran 3 trials per task for  
 94 each model. Open-source baselines (QWEN-480B, GPT-OSS-120B, GPT-OSS-20B) achieved only  
 95 25–50% success, often failing due to hallucinated or invalid ReaScript functions and mis-indexed  
 96 parameters. GPT-5, by contrast, reached 83% success, benefiting from broader ReaScript coverage  
 97 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.

98 **3.3 Subjective Evaluation (MOS Test)**

99 We conducted a **Mean Opinion Score (MOS)** test with 21 participants, who rated DAWZY on a  
 100 5-point Likert scale across five dimensions: **Usability**, **Control**, **Learning**, **Collaboration**, and  
 101 **Enjoyment**. All dimensions scored above the neutral threshold of 3, with *Enjoyment* ( $M = 4.48$ ) and  
 102 *Learning* ( $M = 4.38$ ) rated highest, followed by *Collaboration* ( $M = 4.29$ ), *Usability* ( $M = 4.14$ ),  
 103 and *Control* ( $M = 3.81$ ), reflecting a positive overall perception of the system.

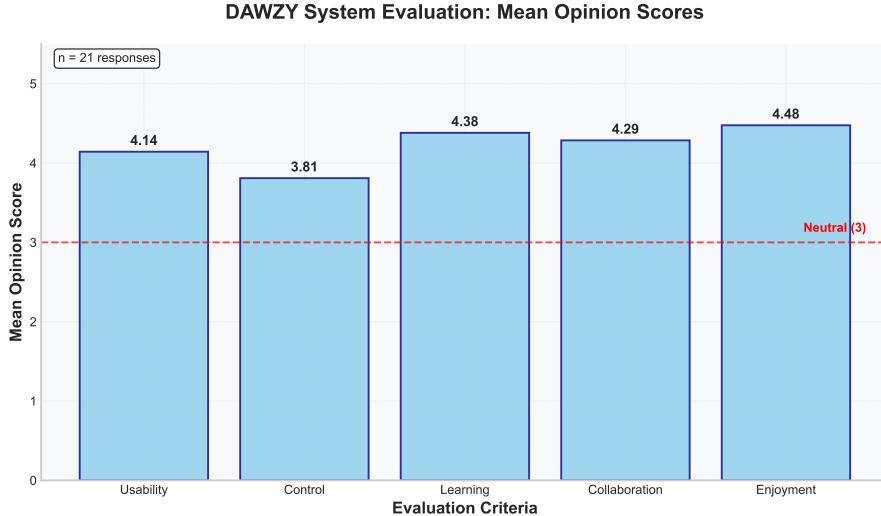


Figure 2: Mean Opinion Score (MOS) results for DAWZY (N=21). The dashed red line indicates the neutral rating (3).

104 **4 Conclusion**

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

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

118 **References**

- 119 Colby N. Leider. *Digital Audio Workstation*. McGraw-Hill, Inc., USA, 1 edition, 2004. ISBN 0071422862.
- 120 Yngvar Kjus. The platformization of music production: How digital audio workstations are turned into platforms  
121 of labor market relations. *New Media & Society*, 2024. doi: 10.1177/14614448241304660. First published  
122 online December 11, 2024.
- 123 Ableton. Ableton reference manual, version 12 — max for live, 2024. URL <https://www.ableton.com/en/manual/max-for-live/>. Accessed 2025-08-18.
- 124 Cockos Incorporated. Reascript documentation. <https://www.reaper.fm/sdk/reascript/reascript.php>, 2024. Accessed 18 Aug 2025.
- 125 S. Bricard, F. D'Errico, M. Devis, A. Bitton, D. ADC, and E. Vincent. Juice: A framework for co-creative agents  
126 in digital audio workstations. arXiv:2402.19323, 2024. URL <https://arxiv.org/abs/2402.19323>.
- 127 Suno. Suno. <https://suno.com/>, 2024. Accessed 18 Aug 2025.
- 128 Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, and et al. Evaluating large language models trained on  
129 code. *arXiv preprint arXiv:2107.03374*, 2021. URL <https://arxiv.org/abs/2107.03374>.
- 130 Anthropic. Introducing the model context protocol. <https://www.anthropic.com/news/model-context-protocol>, 2024. Accessed 18 Aug 2025.
- 131 Xinyi Hou, Yanjie Zhao, Shenao Wang, and Haoyu Wang. Model context protocol (mcp): Landscape, security  
132 threats, and future research directions, 2025. URL <https://arxiv.org/abs/2503.23278>.
- 133 Mozart AI. Mozart ai — ai-powered music production daw, 2025. URL <https://getmozart.ai/>. Product  
134 site; Accessed 2025-08-18.
- 135 OpenJS Foundation. Electron. <https://www.electronjs.org/>, 2024. Accessed 18 Aug 2025.
- 136 Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust  
137 speech recognition via large-scale weak supervision. *arXiv preprint arXiv:2212.04356*, 2022. doi: 10.48550/  
138 arXiv.2212.04356. URL <https://arxiv.org/abs/2212.04356>.
- 139 Spotify. Basic pitch — about. <https://basicpitch.spotify.com/about>. Product page; accessed 18 Aug  
140 2025.
- 141 OpenAI. Introducing gpt-5. <https://openai.com/index/introducing-gpt-5/>, 2025a. Accessed 18 Aug  
142 2025.
- 143 OpenAI. Gpt-5 system card. <https://openai.com/index/gpt-5-system-card/>, 2025b. Accessed 18 Aug  
144 2025.
- 145 Qwen Team. Qwen3 models — hugging face collection. <https://huggingface.co/collections/Qwen/qwen3-67dd247413f0e2e4f653967f>. Collection page; accessed 18 Aug 2025.
- 146 Qwen Team. Qwen3 technical report, 2025. URL <https://arxiv.org/abs/2505.09388>.
- 147 Meta AI. facebook/musicgen-small. <https://huggingface.co/facebook/musicgen-small>. Hugging  
148 Face model card; accessed 18 Aug 2025.
- 149 Jade Copet, Felix Kreuk, Itai Gat, Tal Remez, David Kant, Gabriel Synnaeve, Yossi Adi, and Alexandre Défossez.  
150 Simple and controllable music generation, 2023.
- 151 Rachel Bittner, Min Kim, Juan José Bosch, Aren Jansen, Gordon Wichern, and Longshaop Hantrakul. A  
152 lightweight polyphonic pitch transcription model. *arXiv preprint arXiv:2203.09893*, 2022. URL <https://arxiv.org/abs/2203.09893>.
- 153 Ableton AG. Ableton live 12. <https://www.ableton.com/>, 2025. Accessed 18 Aug 2025.
- 154 Mamdouh Alenezi and Mohammed Akour. Ai-driven innovations in software engineering: A review of current  
155 practices and future directions. *Applied Sciences*, 15(3):1344, 2025. doi: 10.3390/app15031344. URL  
156 <https://www.mdpi.com/2076-3417/15/3/1344>.
- 157