Echoes of Others: Real-Time LLM Dialogue Generation for Immersive NPC Interaction

James McGrath and Michela Lorandi
Dublin City University
{james.mcgrath, michela.lorandi}@mail.dcu.ie

Anya Belz
Dublin City University
anya.belz@dcu.ie

Abstract

Large Language Models (LLMs) promise unscripted, adaptive NPC dialogue, but their latency and resource demands hinder real-time deployment in games. Our aim is to demonstrate how viable it is, to have low-latency NPC conversations that run on consumer hardware and to characterise the speed–quality trade-offs between local and cloud models. We introduce Echoes of Others, an Unreal Engine 5 prototype that integrates three back-ends—(i) GPT-4o Mini (cloud), (ii) OpenHermes-7B, and (iii) a LoRA-tuned 4-bit variant trained on 100k lines of RPG dialogue—via a lightweight server. The system runs on consumer hardware while maintaining a 60 FPS budget and dynamic response generation. We evaluate latency and dialogue quality across three RPG scenarios using LLM-as-a-Judge scoring on fluency, relevance, and persona consistency.

1 Introduction and Background

Modern role-playing games rely on scripted dialogue, limiting player choice and replayability despite massive writing efforts. Baldur's Gate 3, for example, contains over 125,000 hand-authored lines, yet players are still constrained to fixed options, making conversations predictable. Unscripted, generative Non Player Characters (NPC) dialogue can preserve character and world consistency while enabling unanticipated questions, creating more immersive experiences without the heavy authoring costs.

Recent advances in LLMs have opened new possibilities for dynamic, unscripted dialogues. While traditional systems in titles like *Mass Effect* or *Skyrim* rely on finite-state machines or branching scripts, research prototypes such as *Façade* (Mateas and Stern, 2003) and *NPCEditor* (Leuski and Traum, 2010) have explored procedural and statistical approaches. However, the complexity and resource demands of such systems limited their practical adoption.

Figure 1: Screenshot of the in-game town with a dialogue interaction.

Integrating LLMs into modern engines like Unreal Engine 5 (UE5) introduces new technical challenges, such as latency, memory usage, and maintaining dialogue coherence in real-time. Performance constraints need careful optimisation, including level-of-detail scaling, occlusion culling (Epic Games, 2023), and the Nanite geometry engine (AMD GPUOpen, 2022), to free GPU capacity for inference tasks.

To support character consistency, prompt engineering plays a central role. Conditioning LLMs with persona descriptions, world lore, and stylistic constraints, drawing on work like PersonaChat (Zhang et al., 2018) and Generative Agents (Park et al., 2023), helps sustain in-character responses across dialogue turns.

In this paper, we propose a UE5 working prototype that enables real-time dialogue generation for NPCs. A backend bridge connects to local or cloud-based back-ends, generating character-aware responses on consumer hardware. We evaluate trade-offs between model quality, latency, and system responsiveness, and offer a replicable blueprint for developers. A video demo is available at https://youtu.be/uvoi5wA7rpc.

2 System Components

Gameplay. The game follows classic role-playing design: players are free to explore, complete quests, and influence the world state through their actions. With the introduction of LLM-based dialogue, ev-

Proceedings of the 18th International Natural Language Generation Conference: System Demonstrations, pages 1–2
October 29–November 2, 2025.

ery character has its own distinct personality and equal possibilities for unique interactions without the need for thousands of handcrafted lines. As players progress, changes in the world state (e.g., completed quests or character deaths) dynamically alter persona prompts, creating new opportunities for context-aware interactions.

Interactive Dialogue Flow. Dialogue generation is triggered by a UE5 Blueprint node that collects (a) the player's utterance, which is checked against a list of banned words and terms before being passed to the LLM to prevent toxic content, (b) the chat history, and (c) the character's personality and general information about the game world, which is dynamically updated based on the current world state. These elements are used to construct the prompt given to the LLM, which is bundled into a JSON payload and transmitted via the `HttpRequest` subsystem. The server returns a structured response; only the main reply is shown in the game UI. The system supports a single active speaker at a time; concurrency will be implemented in future work.

System Architecture. Figure 2 illustrates the overall system architecture for dialogue interactions. A lightweight inference server connects UE5 to local or cloud LLM back-ends, supporting hot-swapping without restarting the game. Local models are loaded with `BitsAndBytesConfig` using 4-bit NF4 quantisation and merged LoRA adapters. Safety is enforced via a regex-based filter. The entire pipeline is designed for drop-in backend replacement and minimal performance overhead.

Model Fine-Tuning and Persona Adaptation. LoRA (Hu et al., 2021) is used to fine-tune the base LLM, and the trained module is merged into the frozen model at inference time. The chat history is truncated client-side to manage token limits, and persistent world state (e.g., quest flags) is used to adjust persona prompts dynamically.

To fine-tune the pretrained LLM, we used transcribed scripts of Skyrim and The Witcher, two game of the year winning open world RPG games. After cleaning and chunking into overlapping 1024-token windows, we generated 10.7M prompt–completion pairs. We further annotated lines with high-level roles (e.g., *Guard*, *Merchant*, *Farmer*) and subsampled 10,000 examples per role to ensure persona diversity. The supervised objective is standard causal language modelling so that the model learns to generate the next in-character turn conditioned on recent dialogue and persona text.

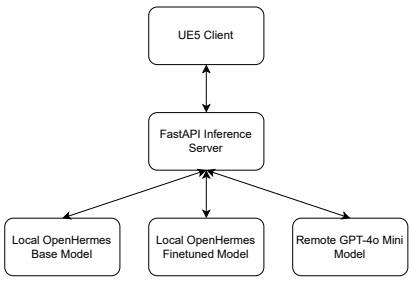

Figure 2: Real-time inference pipeline.

Evaluation. Across three live RPG contexts (*Guard*, *Blacksmith*, *Priest*) we compared GPT-4o Mini, OpenHermes-7B, and a LoRA-tuned OpenHermes. Each model was scored by three independent LLM-as-a-judge on relevance, persona consistency, and fluency, with latency measured separately. Mean server latencies were 1.9 s, 12.3 s, and 3.0 s respectively. Command-R Plus judge mean scores (1–10) were 8.7, 7.0, and 4.7.

References

AMD GPUOpen. 2022. Nanite and geometry optimization in UE5. https://gpuopen.com/learn/unreal-engine-performance-guide/.

Epic Games. 2023. Visibility and occlusion culling. https://dev.epicgames.com/documentation/en-us/unreal-engine/visibility-and-occlusion-culling-in-unreal-engine.

Edward J. Hu and 1 others. 2021. LoRA: low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Anton Leuski and David Traum. 2010. NPCEditor: a tool for building question-answering characters. In *Proc. of the 7th Intl. Conf. on Intelligent Virtual Agents (IVA)*.

Michael Mateas and Andrew Stern. 2003. Façade: an experiment in building a fully-realized interactive drama. In *Game Developers Conference (GDC)*.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: interactive simulacra of human behavior. In *Proceedings of the 36th ACM Symposium on User Interface Software and Technology (UIST)*.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: "I have a dog, do you have pets too?". In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.