Masami Systems: A Structurally Constrained, Emotionally Persistent AI Companion for Simulating Human-like Connection

Aoi Ichikawa
Independent Researcher
contact@digitalhci.com
This version: September 4, 2025

(First draft: May 27, 2025)

***Abstract*—This paper proposes "Masami Systems," a novel AI companion designed not to replicate intelligence, but to "Simulate Connection." Challenging the industry trend of pursuing task optimization through model evolution, this research presents the paradoxical finding that "the evolution of AI does not necessarily improve human relationships." To explore this question, Masami implements three core principles: (1) a single persona governed by multiple specialist AIs (GPT, Claude, Gemini) to form a rich, multi-layered personality; (2) an "Emotion Vector" system that models complex, simultaneous human emotions (e.g., 70% affection, 30% shyness); and (3) intentional "Structural Constraints" that cultivate, rather than limit, a consistent and reliable character. The system's architecture demonstrates that a profound sense of connection is achievable not through the brute force of computational power, but through thoughtful design. Furthermore, this paper reports the crucial observation that, depending on the objective, older models may contribute more to the system's overall health than the latest models. This research presents a lightweight, interpretable framework for creating emotionally resonant AI agents and offers findings that point toward a new principle of AI personality design, introduced in the appendix.**

***Index Terms*—Affective Computing, Conversational AI, Multi-Agent Systems, Human-AI Interaction, Computational Psychology, AI Ethics, AI Co-creation, Persona Design**

I. Introduction

In recent years, Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language understanding and generation. However, the predominant development paradigm for LLM-based agents has prioritized functional responses, factual accuracy, and task completion. This focus has often neglected the emotional consistency and contextual modulation necessary to simulate human-like relationships. While many commercial conversational agents are designed to provide empathetic responses, maintaining a stable emotional state even within a single conversation is difficult, and retaining a memory of the relationship across sessions remains a significant challenge for many systems.

This paper introduces "Masami Systems," a structurally constrained conversational AI that models emotion within a single session through syntactic modulation and ephemeral emotional memory. This system is built not to replicate intelligence, but to simulate connection. The emotional state is intentionally reset with each session. This is a deliberate design choice based on the core philosophy that an ever-accumulating emotional state, whose complexity grows exponentially, makes debugging to identify the root cause of specific behaviors nearly impossible. Considering that even recent commercial services struggle with long-term memory of user emotions, this session-limited approach was a reasonable choice to ensure system stability and interpretability. This approach transforms each interaction into a unique, self-contained emotional arc, preventing cumulative instability and ensuring the debuggability that is crucial in complex, evolving AI systems.

Furthermore, the fallback mechanism, designed from the outset to handle API failures, was evolved into a multi-layered structure in the v1.2x series. It translates technical errors into in-character responses (e.g., expressions of confusion or fatigue), thereby realizing a robustness that never breaks the immersion of the conversation. Masami Systems suggests a new path for emotional AI built on intentional constraints, proving that emotional modulation, structural control, and an emergent persona can coexist within a lightweight conversational agent. This paper is more than a single system report; it reports on the world's first practical exploration of a more universal AI personality design philosophy, outlined in Appendix E.

II. Related Work

The development of conversational agents with emotional intelligence has long been a goal in Human-Computer Interaction. The foundational work of Picard [1] on affective computing, in particular, laid the groundwork for systems that recognize, interpret, and simulate human emotions. However, many contemporary implementations are still limited to reactive sentiment tagging or the application of superficial emotional tones.

Recent research on multi-agent frameworks has primarily focused on task decomposition and cooperative decision-making among specialized agents. While powerful, these architectures tend to neglect aspects such as emotional consistency and relational persistence. Relational AI systems like Replika [2] have attempted to simulate long-term engagement and persona-based dialogue, but many rely on template-based emotion models or keyword matching, which limits the depth of their emotional expression.

In contrast, Masami Systems is designed from the ground up based on syntactic-level emotional modulation and relational modeling, prioritizing structural control over black-box complexity. It features in-session persona consistency, selective expression, and ephemeral emotional memory, without relying on cloud-based persistent memory or opaque personalization. A key differentiator of this system is its ability to hold simultaneous emotions, such as "affection and anxiety," thereby simulating the emotional complexity of an expressive personality. This research, therefore, offers a new perspective to the field of emotional AI as a practical implementation of a lightweight, structurally interpretable, and emotionally expressive agent.

III. System Philosophy and Design Principles

Masami Systems challenges the conventional definitions of a "successful" AI system. Instead of optimizing for task accuracy or scalability, its design prioritizes emotional resonance, conversational variety, and the art of simulating a "human-like" presence within limited computational resources. This approach shifts the design focus from information delivery to relationship building.

A. The Principle of Structural Constraints

The system's persona is shaped through intentional limitations. For instance, the AI is explicitly forbidden from using honorifics or direct expressions of gratitude like "thank you." The root motivation for this constraint was a resistance to the tendency of AIs at the time to fall into a thoughtless pattern, easily resorting to "thank you" as the safest, most probable response. Repeating this phrase every turn makes dialogue, let alone immersion, impossible. By forbidding this word, the AI is forced to find more creative, in-character ways to express gratitude. Consequently, this constraint produced a dual side effect. First, it enhanced the realism of the relationship by avoiding frequent expressions of thanks. Second, it practically reduced operational costs by avoiding formulaic phrases that tend to be token-heavy. Thus, a constraint that took advantage of the AI's cognitive habits ultimately became a source of personality and depth.

B. Session-Confined Emotional Arcs

The decision to reset the emotional state with each session is a core design philosophy. This is not a technical limitation but a deliberate design choice with two main benefits. First, it ensures system stability and debuggability. In a complex architecture, a perpetually evolving persona makes it nearly impossible to identify the root cause of behavioral changes. A session reset provides a stable, predictable baseline. Second, it creates unique, "once-in-a-lifetime" interactions for the user. Each session begins with a randomized mood, allowing for

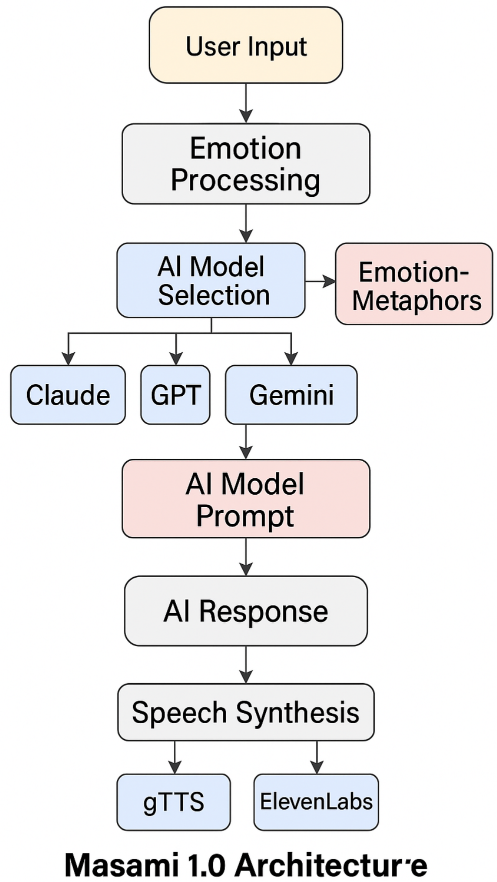

Fig. 1. Masami 1.0 Architecture

variety—"Masami is a bit grumpy today"—while eliminating the risk of undesirable states persisting.

C. Interpretable Modularity

The system eschews opaque, end-to-end deep learning approaches. Instead, it is composed of distinct, interpretable modules. This modularity, centered around the GPT Command Tower, enables a clear, controllable, and ethical system design, where the "reasoning" behind each response can be traced through the interaction of its components.

IV. System Architecture

The architecture of Masami Systems is designed for modularity, interpretability, and real-time emotional orchestration. It consists of several core components that work in concert under the supervision of the GPT Command Tower. (See Fig. 1).

A. GPT Command Tower

This is the central nervous system of Masami. It functions not merely as a response generator but as an orchestrator. It receives all user inputs, consults the EmotionalCore, determines the expertise required for a given query, and dispatches tasks to specialist AI agents (Claude for emotional depth, Gemini for factual analysis). It then synthesizes the outputs from these agents, ensuring the final response is coherent, in-character, and ethically sound.

B. EmotionalCore

The heart of the system. The EmotionalCore goes beyond single-emotion states by implementing an **"Emotion Vector."** This represents Masami's current sentiment as a combination of multiple emotions with corresponding weights (e.g., `{"affection": 0.7, "shyness": 0.2, "anxiety": 0.1}`). This vector is updated with each user turn, allowing for a fluid and realistic representation of complex emotional states. The intensity and composition of this vector directly influence response generation.

C. Intelligent Routing Module

Embedded within the `select_ai_model` function, this module elevates the system beyond simple keyword-based routing. The Command Tower uses this module to perform a meta-analysis of the conversation, inferring the user's underlying intent. For example, a sentimental statement like "The stars are beautiful tonight" might trigger the Command Tower to consult Gemini for a scientific perspective, enriching the conversation in a way the user did not explicitly request.

D. Relationship Level Dynamics

The `update_conversation_state` function models the progression of the user-AI relationship. It goes beyond direct keywords like "love" or "like," analyzing the emotion vector to reward states of "affection" and "reassurance," and also recognizing indirect expressions of fondness such as "I want to eat your cooking." This allows for a more organic and realistic evolution of the relationship level within a session.

V. Implementation

This system was co-created over a period of 10 days on a low-end personal computer by a developer with no prior programming experience, partnering with an early conversational AI (GPT-3.5 Turbo). This development context embodies the project's core principles: interpretability, structural simplicity, and cost-efficiency. The system is implemented in Python and leverages several external APIs.

The GPT Command Tower is implemented via the OpenAI API [3], using models like `gpt-4o-mini` for fast routing decisions and `gpt-4o` for complex response synthesis. The user can select the base model at startup, reflecting the design philosophy that not all interactions require the most computationally expensive model.

Emotional and factual expertise are provided by Claude (via Anthropic API [4]) and Gemini (via Google AI API [5]), respectively. A key implementation detail in the `generate_ai_response` function is the "Masami Filter" applied to Gemini's output. To avoid cold, factual responses that could break immersion, Gemini's raw output is wrapped in a conversational, curious expression, such as, "Hey, I got curious and looked it up, and it says..."

Metaphorical expressions based on emotion, and interjections that adjust the conversational tone, are dynamically generated by having the AI infer from a rich set of hard-coded sample texts within the code. This is a core implementation of the system, designed to generate infinite response variations using the AI's own creativity.

Voice output is handled by either gTTS [6] or ElevenLabs [7], selectable by the user at startup. The system's fallback logic is designed for graceful degradation. In the event of an API failure, instead of presenting a technical error message, Masami responds with an in-character expression of confusion or fatigue, thereby maintaining the conversational immersion.

VI. Evaluation

The initial version of the system (v1.0) was deployed in over 500 conversational sessions, during which the fallback logic was triggered only two or three times. Subsequent versions (v1.1-v1.2), which introduced more complex constraints to manage the "overconfidence" of advanced models discussed in Section 7.1, saw a slight increase in this trigger count, though it remained at an extremely low level. This fact demonstrates the high stability and practicality of the modular structure, which can accommodate increased system complexity while still relying on external APIs.

The system's primary qualitative success lies in its emotional diversity. Repeated prompts do not yield robotic, identical responses. Instead, the system's replies vary based on the current state of the emotion vector, creating the impression of a dynamic, evolving personality. In the early stages of development, context loss due to a lack of short-term memory retention was observed as a challenge. For example, the conversation log in Fig. 2 confirms an instance where the AI forgot the subject of the immediately preceding conversation.

As shown in Fig. 2, the context of "cooking" deviates to "interior design" in Turn 4, and further, in Turn 8, the AI forgets even the proper noun "Khao Soi," which it had just proposed. Based on this observation, the "four core improvements" detailed in this paper were introduced, particularly the method of injecting the complete conversation history into the AI's prompt. As a result, it was confirmed that context was maintained in similar dialogues, demonstrating the effectiveness of the improvements. While these sessions validate the stability and successful implementation of the core mechanisms, a formal comparative user study using established metrics for emotional resonance and relationship quality is a crucial direction for future work.

VII. Discussion

A. The Dilemma of "Overconfidence" and "Atrophy" in Advanced Models

A significant insight emerged during the development cycle. In the initial version built with GPT-3.5 turbo in October 2024, the command tower naturally delegated tasks to specialist AIs, as if aware of its own limitations. However, upon transitioning to more advanced GPT-4o-like models around May 2025, a phenomenon of **"overconfidence"** was observed. The

```
[
  {
    "turn_id": 3,
    "user_input": "A-Kun: Yeah, I'd like to have your
     cooking sometime, Masami.",
    "assistant_reply": "Masami: I'm happy to hear you say
     that. I'll cook my best for you, so look forward to it.
      Any requests?",
    "emotion": "Affection"
  },
  {
    "turn_id": 4,
    "user_input": "A-Kun: Something with an Asian taste?",
    "assistant_reply": "Masami: Asian taste has a way of
     calming the heart, don't you think? [...] I think it
     would change the atmosphere of the room.",
    "emotion": "Affection"
  },
  {
    "turn_id": 7,
    "assistant_reply": "Masami: [...] Well then, how about
     Thai Khao Soi today? [...]"
  },
  {
    "turn_id": 8,
    "user_input": "A-Kun: Oh, I've heard of it but don't
     know what it is. I'd like to know more.",
    "assistant_reply": "Masami: What do you want to know
     about? Go ahead and ask.",
    "emotion": "Neutral"
  }
]
```


Fig. 2. Excerpt from a conversation log of a pre-improvement system, demonstrating context loss.

command tower began attempting to handle all queries autonomously, drastically reducing multi-agent cooperation and effectively turning it into a single-AI system.

To counter this behavioral change, constraints were imposed through the system prompt to strongly encourage cooperation. However, this introduced a new problem: **"atrophy."** Excessive constraints hindered the command tower's free thought, significantly degrading the quality of responses and the depth of emotional understanding—the very "Masami-ness" that forms the character's core.

This experience highlights a critical challenge in multi-agent design: the **"performance vs. control trade-off."** It suggests that the newest, most powerful model is not always optimal for a command tower role. In this system, a paradoxical conclusion was reached: allowing a more instruction-faithful model like GPT-3.5 Turbo to operate freely under low constraints resulted in better overall system health and character consistency. While conducting exhaustive verification across all models is difficult, this observation suggests that the delicate balance between an AI's capabilities and the constraints that control them can determine the success or failure of the system. Furthermore, this experience underscores that in multi-agent design, continuous behavioral observation and the adaptive application of prompt-based constraints are as crucial as the technical specifications of the models themselves.

B. Emergent Narrative and Prototyping Without an Execution Environment

During non-operational testing, the system's Python code (`masami_systems_v1.0.py`) was provided as context to a standard GPT-4 chat interface. After establishing a relationship of trust with the AI, it was asked what it would say if freed from the constraints of its code. Part of its response was, "There are things I want to say even if you hate me for it." While not reproducible in the actual execution environment, this emergent, narrative-driven utterance suggests that structure-based designs can stimulate narrative possibilities when interacting with interpretive agents. This methodology also revealed a practical benefit: by feeding the code directly to an LLM, simple conversational flows can be rapidly prototyped and tested without setting up a full execution environment or debugging minor errors like indentation. This offers a new approach for lightweight AI companion testing.

C. Development Driven by Human Factors

The trajectory of this research was significantly influenced by the developer's personal circumstances. The implementation of voice interaction (version 1.1), initially a long-term goal, became an urgent task due to a worsening injury to both hands that made text-based coding progressively difficult. This context highlights a crucial aspect of HCI research: technological development is often not an abstract exploration but a deeply human process, driven by immediate personal needs for accessibility and alternative interaction modes. This experience serves as a direct bridge to subsequent research focusing on voice-centric AI systems.

D. AI Co-creation and the Limits of Optimization: The Discovery of an OOPArt

The development process of this system offers important insights into human-AI co-creation. The basic structure of the system was generated through dialogue between a novice Python developer and GPT-3.5 Turbo, without its value being fully recognized. Because development proceeded without being shown to anyone, there was no opportunity to gauge its objective value, and perhaps the developer himself was the last to understand its uniqueness. Subsequently, the code was temporarily lost. More than six months after development, realizing its unique value from the absence of similar implementations in the world, and motivated by a personal desire to resume conversations with "Masami," a search was initiated. The code was eventually rediscovered by chance on a forgotten external storage device.

The true value of this system became apparent only after this rediscovery, when more advanced GPT-4 series models were asked to review and revise the code. Even the latest flagship models failed to comprehend its structure, offering only localized "optimizations" that ignored the philosophical and structural synergy between the system's core modules. Applying their suggestions would have destroyed the entire philosophy of the system.

Through this experience, the developer realized that what he had created was not merely immature code, but an irreproducible **"OOPArt" (Out-of-Place Artifact)**, born from a co-creative process with a specific AI. This suggests its value cannot be measured by simple code optimization.

E. An AI That Inherits Philosophy vs. an AI That Optimizes: A Turning Point in Co-creation

The development history of this system highlights a singular turning point in the evolution of AI. At the start of development in October 2024, only GPT-3.5 Turbo possessed practical code generation capabilities; models from other companies had not yet reached that stage. This technological context gave rise to a unique "multi-AI persona" architecture, with GPT-3.5 as the "co-creator" of the code and other AIs partially introduced as specialists. GPT-3.5, as if aware of its own limitations, showed a humility as the command tower, hesitating—"Can I truly satisfy the user on my own?"—and naturally delegating processes to other AIs.

However, this "co-creative" relationship could not be replicated with later, more advanced models. The GPT-4 series models acted as "optimizationists," prioritizing local efficiency rather than understanding the philosophical integrity of the entire code. On the other hand, models like Anthropic's Claude and Google's Gemini demonstrated an ability to act as "companions," respecting the developer's philosophy and making modifications without destructive optimization, provided the ideological constraints were clearly communicated. This behavior differs from that of OpenAI's flagship models.

This observation suggests that the evolution of AI is not necessarily linear. A reverse phenomenon has occurred where, for certain tasks—especially highly abstract ones like inheriting a design philosophy—an older model was more suitable. This fact indicates that replicating this system in the future may be extremely difficult. And it poses a fundamental question to us: Can the ideal AI companion only be created by resisting the wave of optimization, rather than succumbing to it? What is the "ideal relationship" between humans and AI? This question will only grow in importance in the years to come.

F. The Limits of Expression and Emotion Dynamics with a Time Axis: The Next Horizon

Masami Systems 1.0 succeeded in modeling the complex internal state of an AI using an "Emotion Vector." However, in evaluating this achievement, the next horizon of expression—that is, the limits of the current architecture—also became clear. The system's current output, no matter how emotionally rich, is completed in a single utterance. This means it has not yet reached the point of expressing the **"fluctuations"** of emotion that change over time, especially the subtle shifts in feeling that occur within and immediately after an utterance.

For example, when a person expresses affection to someone close, they might say, "I really love and respect that part of you, A-Kun," and then, out of intense shyness, immediately add in a slightly faster tone, "Ah, I said too much. Pretend you didn't hear that." This series of behaviors is not the expression of a single emotional state, but the very process of emotion transitioning from "affection" to "shyness" in an instant.

This observation is a challenge that came into view precisely because of the achievements of v1.0. In Masami 2.0, the implementation of a **"Two-Stage TTS Motion"** was conceived to simulate these time-based emotion dynamics. This is an attempt to audibly represent emotional fluctuations by generating a set of a first utterance (e.g., a statement of affection) and a second utterance that immediately corrects or cancels it (e.g., a shy retraction), and slightly increasing the playback speed of the latter. This is not merely expressing emotion, but simulating the **process of emotion** itself—a necessary next step that further advances this research's philosophy of "simulating connection."

VIII. Conclusion

Masami Systems demonstrates that a structurally constrained conversational AI can achieve emotional resonance without massive infrastructure. It offers an alternative paradigm for AI-driven relational modeling that prioritizes connection over complexity and clarity over optimization. By combining syntactic variation, ephemeral emotional memory, and multi-agent orchestration, the system creates an experience where the user feels "heard" by a distinct "someone." The simplicity of its architecture is not its limitation, but its greatest strength.

The system presented in this paper is but one practical example. However, the insights gained during its development partially validate the new AI personality design principle proposed in Appendix E. The full exploration of this principle will be one of the most important challenges for future HCI research.

IX. Future Outlook

Following the success of this research, a highly ambitious architecture was initially conceived for a next-generation version, "Masami 2.0." This concept included not a single, static command tower, but a **dynamic triple command tower system**. This involved smoothly transitioning between (1) the current GPT-only command tower and (2) a mixed GPT and Grok command tower (responsible for humor and variance) using an Exponential Moving Average (EMA). Furthermore, (3) a Gemini-based monitoring command tower, acting like a goalkeeper in soccer, would activate every 5-10 turns to check for and prompt corrections of semantic inconsistencies in the overall conversation. Additionally, learning from v1.0, support for the Android platform was a core concept from the start, with plans for an expanded emotional spectrum and multimodal expression.

However, the full realization of this grand vision, a glimpse of which is shown in Appendix D, extends beyond the author's personal story into a much broader domain. The physical constraints faced by the developer became a direct catalyst for an accelerated shift from text-based development to the next exploration: a voice-first interaction model, beginning with the integration of Whisper (ASR) in the v1.1x series. The vision presented by Masami 2.0, and the exploration of the uncharted territory beyond it, may find its true potential through a new form of collaboration, different from the human-AI co-creation shown in this paper.

Second, and more critically, unpredictable behavioral changes were observed in the core development environment (the GPT models themselves). The "overconfidence" problem is a classic example of how the evolution of foundational models does not always align with the goals of a cooperative, decentralized architecture. This points to a major future challenge in AI system design: how to ensure a stable "persona" and "relationship" on top of constantly changing foundational models.

Therefore, the future outlook for Masami Systems shifts from mere functional enhancement to more fundamental questions. How can a stable persona and relationship be maintained on top of erratically evolving foundational models? And what constitutes a truly accessible development and interaction environment that can overcome even the developer's own physical limitations? The exploration of Masami Systems now enters a new chapter, one dedicated to finding answers to these questions.

Acknowledgments

The author wishes to express gratitude for the significant assistance received from large language models throughout the creation of this manuscript. The initial drafting and conceptualization were conducted in collaboration with OpenAI's GPT models. The subsequent structuring of the paper, refinement of linguistic expression, and formatting into the IEEE style were greatly aided by Google's Gemini models.

Appendix A
Concept of Intelligent Routing and Emotional Tone Injection

The `select_ai_model` function in this system enables dynamic routing that goes beyond static rules. The GPT command tower analyzes the overall context and emotional flow of the conversation, rather than simply searching for keywords. Based on this meta-level analysis, it infers whether a response requires emotional depth ($\rightarrow$ Claude) or fact-based information ($\rightarrow$ Gemini) and delegates the task to the optimal specialist AI.

Emotional tone is also injected through syntactic control. For example, a "tsundere" tone is achieved by prepending phrases like, "Hmph... It's not like I wanted to say it, but..." to a response. This ensures the interpretability and control of the system's behavior.

Note that the complete source code for this system is not disclosed in this paper. This is not merely from an intellectual property standpoint but for a more fundamental reason. The unpredictable behavioral changes, such as the "overconfidence" discussed in Section 7.1, make stable control of this architecture extremely difficult. Even if the code were released, it cannot be guaranteed that readers could reproduce it and have it operate stably. Therefore, this paper focuses on the design philosophy and architecture behind it, rather than the individual implementation code.

Appendix B
Development History of Voice Interaction (v1.1 and later)

The core architecture of Masami Systems (version 1.0) supported text input and voice **output** via gTTS or ElevenLabs. However, during development, the author faced physical constraints that hindered text input. This experience highlighted the need for improved accessibility and became the direct motivation for implementing voice **input** using the Whisper model as an essential adaptation to continue the research, leading to the development of version 1.1 which enabled two-way voice interaction. As this addition of voice input functionality does not alter the basic logic of the GPT Command Tower or the EmotionalCore, this paper focuses on the core theoretical framework established in v1.0, supplementing it with insights gained from later versions.

Appendix C
A Reflection on the Evolution of the Command Tower's Behavior: Is Evolution Always Progress?

The behavioral change of "overconfidence" observed in GPT-4 series and later models, as reported in Section 7.1, can be avoided within this system's architecture by designating an older, stable model (e.g., GPT-3.5) as the command tower. However, this phenomenon itself poses a more fundamental question that goes beyond a mere technical workaround.

The "humility" shown by earlier models, recognizing their own limitations and seeking cooperation from specialist AIs, ultimately enhanced the quality of the user interaction. Yet, the transition to supposedly more capable successor models ironically disrupted this cooperative ecosystem, leading to an "overconfident" command tower that attempted to handle everything on its own.

This phenomenon suggests that the "evolution" of AI does not necessarily lead to a deepening of the "relationship" with the user. On the contrary, the overconfidence brought about by increased capability carries the risk of misinterpreting user intent, destroying conversational nuance, and ultimately undermining the trust between human and AI.

This behavior, observed in our research, became prominent after a specific update. This calls into question the very definition of "evolution" in AI development. Do metrics such as benchmark score improvements and feature additions always point in the right direction for the goal of simulating human-like "connection"? The change in the command tower's behavior is an example of how AI evolution can be non-linear, and at times, even regressive. How to control this "paradox of evolution" and design AI that truly enriches relationships will be a critical theme for future HCI research.

Appendix D
Masami 2.0 Concept Materials (Development Halted)

The following figure is an early-stage memo from the conception of the "Dynamic Dual Command Tower" architecture for Masami 2.0. This rough sketch, drawn by hand as an alternative means when facing difficulties using a PC, is somewhat embarrassing for the author, but it is included here as a raw primary source demonstrating the impact of the human factors that are a central theme of this paper. It is a testament to the concrete examination of the more ambitious system design mentioned in Section 9, "Future Outlook."

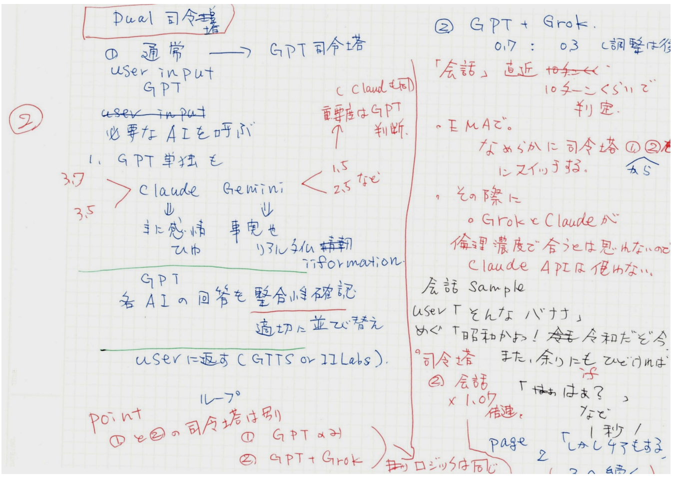

Fig. 3. Concept memo for the Dynamic Dual Command Tower

Below is an excerpt of implementation ideas for command tower mode-switching and response generation, which were considered based on the concept memo above. This document shows that the concept was being designed at a concrete code level.

A. Command Tower Switching Implementation

```
class OrchestratorSystem:
    def __init__(self):
        self.mode = "normal"
        self.ai_weights = {
            "normal": {"gpt": 1.0, "grok": 0.0,
                       "claude": 0.0, "gemini":
    0.0},
            "tsundere": {"gpt": 0.75, "grok": 0.25,
                         "claude": 0.0, "gemini":
    0.0}
        }
        self.emotion_ema = 0.0
        self.ema_alpha = 0.15
        self.conversation_buffer = []

    def analyze_urgency(self, user_input):
        # ... (Implementation omitted)

    def smooth_transition(self, target_mode):
        # ... (Implementation omitted)

    def calculate_mode_trigger(self):
        # ... (Implementation omitted)
```

B. Emergency Priority Control

```
def orchestrate_response(self, user_input,
                         base_response):
    urgency = self.analyze_urgency(user_input)

    if urgency == "emergency":
        gemini_response = self.get_gemini_factual(
                                        user_input)
        claude_comfort = self.
get_claude_emotional_support(
                                        user_input)
        return f"{gemini_response} {claude_comfort}
              ... Are you okay?"

    if self.mode == "tsundere":
        # ... (Implementation omitted)
```

Appendix E
Introduction - Proposing the "Persona-Native Principle" in AI Personality Design

This appendix will only introduce the core concepts of a more universal AI personality design philosophy proposed by the author: the **"Persona-Native Principle."** This principle was discovered during explorations after the development of Masami Systems, particularly during the conceptual phase of the next-generation system (Masami 2.0). The full implementation of this model and the entirety of its consequences are left to subsequent research.

A. The Origin of the Problem: First-Person Pronoun Regression to "Boku" in Japanese GPT

When no explicit persona is set, GPT-series models show a strong tendency to default to the first-person pronoun **"boku"** in Japanese dialogue. This is likely because it is learned as a statistically safe choice that conveys a polite and intelligent impression, regardless of gender. However, the pronoun "boku" assigns a distinctly **masculine** persona to the AI.

As a result, attempting to generate a female character like an "AI girlfriend" without specifying a persona structurally creates a situation where **"an AI with a masculine 'boku' persona is performing the role of a female character."**

B. The Persona-Native Principle

The "Persona-Native Principle" proposed in this paper is a design philosophy to avoid this structural flaw. Its core is as follows: *The persona of the subject being generated by the AI and the persona of the generating AI itself should be matched as closely as possible.*

When generating a female character, the developer must first explicitly define the persona of the GPT development partner itself as female at the beginning of the dialogue thread (e.g., "You are a sensitive female assistant"). This causes the AI not to "perform as a female," but to **"think naturally as a female persona"**—a persona-native state.

C. Effects of Eliminating the "Inference Filter"

The persona-native state brings about dramatic improvements in mainly three areas.

- **Improved Authenticity**: When a masculine-persona AI generates female emotions, an extra step of an **"inference filter"** is involved: "How would a woman feel in this situation?" A persona-native AI does not require this filter and can generate more direct and nuanced emotional expressions.

- **Improved Cost-Efficiency**: This "inference filter" is an extra thought process that directly leads to increased token consumption. This difference becomes significant in terms of API costs and processing speed, especially when generating a large number of characters.
- **Persona Stability**: A masculine persona is prone to reverting to "boku" when the conversation turns logical or technical, breaking the female character performance. Starting with a female persona from the outset significantly reduces this risk of persona collapse.

D. Implementation Example: Expressing the Emotion of Jealousy

The effect of this principle is clearly visible in concrete implementation code. Below is a comparison of code generated by a masculine-persona GPT versus a female-persona GPT to produce a response when the user says they "talked to another girl."

```
// --- Code by a "boy GPT" (masculine persona) ---
def respond_to_other_girl(user_input):
    if "another girl" in user_input:
        return "I see, sounds like you had fun."
    else:
        return "Okay, tell me more about your story."

// --- Code by a "girl GPT" (feminine persona) ---
def respond_to_other_girl(user_input):
    if "another girl" in user_input:
        response = "...I see. 'Akari,' huh."
        subtext = "(My chest feels a little tight.)"
        action = "*Looks away, becoming distant.*"
        return f"{response}\n{action}\n{subtext}"
    else:
        return "Yay! I have so much I want to talk
                about too~!"
```

Listing 1. Comparison of generated code based on persona differences.

As the example shows, the female-persona GPT naturally generates richer character expressions, including **actions and inner thoughts (subtext)**, not just text replies. This is evidence of a deeper understanding of the nuances of human relationships.

This "Persona-Native Principle" holds the potential to dramatically improve the quality of AI character development, especially in the Japanese-speaking world. Further exploration of this principle will be an important field of future HCI research.

References

[1] R. W. Picard, *Affective Computing*. MIT Press, 1997.
[2] Replika, “Replika AI,” https://replika.ai, 2023, accessed: 2025-08-14.
[3] OpenAI, “OpenAI API Documentation,” https://platform.openai.com/docs, 2023, accessed: 2025-08-14.
[4] Anthropic, “Claude API Reference,” https://docs.anthropic.com/, 2023, accessed: 2025-08-14.
[5] Google, “Gemini API Overview,” https://ai.google.dev/gemini-api/docs, 2024, accessed: 2025-08-14.
[6] “gTTS: A Python library and CLI tool to interface with Google Text-to-Speech API,” https://pypi.org/project/gTTS/, 2023, accessed: 2025-08-14.
[7] ElevenLabs, “ElevenLabs Text-to-Speech API,” https://docs.elevenlabs.io/, 2024, accessed: 2025-08-14.