State-Inference-Based Prompting for Natural Language Trading with Game NPCs

Minkyung Kim
SayBerryGames
Daejeon, South Korea
mkkim1778@sayberrygames.com

Junsik Kim
Amazon
Seattle, U.S.A.
jskimcv@amazon.com

Hwidong Bae
xl8
Seoul, South Korea
bae.hwidong@gmail.com

Woongcheol Yang
SayBerryGames
Daejeon, South Korea
woongy@sayberrygames.com

Sangdon Park*
SayBerryGames
Daejeon, South Korea
chaos@sayberrygames.com

Sohee Bae*
SayBerryGames
Daejeon, South Korea
bread@sayberrygames.com

Abstract

Large Language Models enable dynamic game interactions but struggle with rule-governed trading systems. Current implementations suffer from rule violations, such as item hallucinations and calculation errors, that erode player trust. Here, State-Inference-Based Prompting (SIBP) enables reliable trading through autonomous dialogue state inference and context-specific rule adherence. The approach decomposes trading into six states within a unified prompt framework, implementing context-aware item referencing and placeholder-based price calculations. Evaluation across 100 trading dialogues demonstrates >97% state compliance, >95% referencing accuracy, and 99.7% calculation precision. SIBP maintains computational efficiency while outperforming baseline approaches, establishing a practical foundation for trustworthy NPC interactions in commercial games.

CCS Concepts

• **Computing methodologies** → **Discourse, dialogue and pragmatics**; **Natural language generation**; • **Human-centered computing** → *Natural language interfaces.*

Keywords

Large Language Models (LLMs), Game AI, Natural Language Interaction, Prompt Engineering, State-Inference-Based Prompting

1 Introduction

Large Language Models (LLMs) are revolutionizing non-player character (NPC) interactions in games, enabling natural language-driven experiences that transcend traditional script constraints [1, 2]. These capabilities have spurred research and development of dynamic NPCs that enhance player immersion across diverse gaming contexts [3–8]. Commercial implementations already demonstrate this potential in titles such as Relu Games' 'Magical Mic Duel: Senpai, Hear My Spell', 'Uncover the Smoking Gun' and Krafton's 'inZOI' [9–11].

Trading mechanics represent a critical test case for LLM-driven interactions, integrating economic systems, social dynamics, and narrative progression under structured rule systems. Role-playing games particularly depend on merchant interactions as core experiences that bridge exploration, combat, and character development. Yet most games implement trading through static inventories, fixed pricing, and limited dialogue options [12]—constraints that prevent dynamic commercial interactions and limit emergent gameplay potential.

Recent research explores LLM-powered merchant NPCs but typically addresses isolated aspects rather than comprehensive interaction flows. Critical issues persist: item hallucinations, inventory inconsistencies, and rule violations that systematically undermine system reliability [6, 12]. Current LLM-based NPC research emphasizes social interactions [3, 4], leaving rule-governed trading as a fundamental challenge [13].

This work develops a comprehensive framework that balances natural language flexibility with systematic rule adherence for robust trading systems. The framework enables LLMs to infer dialogue states autonomously from the dialogue context without external state management. This is achieved by decomposing trading into six distinct states with comprehensive guidelines within a unified prompt architecture.

The proposed methodology leverages LLMs' generative capabilities while implementing robust control mechanisms to address hallucinations and ensure game system integrity. The approach enhances trading with three core innovations: structured state progression rules preventing accidental transactions, dynamic item reference adjustment based on dialogue context, and placeholder-based post-processing ensuring mathematical accuracy without computational overhead.

Empirical evaluation conducted in an operational role-playing game environment demonstrates the methodology's effectiveness through comprehensive metrics, including rule compliance rates, item reference accuracy, and price calculation precision. The results demonstrate practical viability in commercial game environments.

This work makes the following key contributions:

- This study proposes a State-Inference-Based Prompting (SIBP) methodology that enables autonomous dialogue state inference and rule compliance through four key prompt design elements.
- The methodology decomposes trading into six states, ensuring procedural integrity while preventing unintended behaviors.

*Co-corresponding authors.

KDD '25 Workshop on Prompt Optimization, Toronto, ON, Canada
© 2025 Copyright held by the owner/author(s).

- The evaluation provides empirical validation through quantitative assessment demonstrating over 97% state transition compliance and deployment potential.
- The SIBP methodology facilitates context-aware data referencing mechanisms that achieve over 95% accuracy through dynamic adjustment while preventing hallucinations.
- The SIBP methodology enables state-specific post-processing that ensures 99.7% price calculation accuracy through placeholder-based mechanisms.

The system architecture and prompt design principles offer broad generalizability for LLM-based interactive systems beyond gaming contexts, contributing fundamental advances to prompt engineering for rule-governed interactions. The implications extend to interactive kiosks, customer support systems, and educational tools, demonstrating how sophisticated interaction design can transform LLM capabilities into reliable task-oriented systems suitable for commercial deployment.

The remainder of this paper is organized as follows: Section 2 reviews related work; Section 3 details the proposed methodology; Section 4 provides experimental evaluation; Section 5 discusses significance and implications; Section 6 concludes with contributions and future directions.

2 Related Work

LLMs have transformed interactive entertainment, creating new opportunities for dynamic player experiences [1, 2]. Traditional NPCs operate through pre-written dialogues and finite state machines, resulting in predictable interactions that reduce immersion. Recent research explores LLM-driven approaches for naturalistic and adaptive NPC dialogues, demonstrating how language models can drive agents with sophisticated social interactions and persistent memory capabilities.

Park et al. pioneered Generative Agents, demonstrating that LLMs can generate believable human-like behavior through integrated long-term memory, reflection, and planning mechanisms [3]. Marincioni et al. investigated how emotional states expressed by LLM-based NPCs affect player responses using emotion extraction methods [14]. Recent studies have also examined NPC interactions in VR environments, comparing different communication methods with LLM-based characters [15]. However, these studies focus primarily on open-ended conversations rather than structured, goal-oriented interactions requiring strict rule adherence.

Several studies have examined commercial interactions and negotiation dynamics using LLMs. Abdelnabi et al. proposed a benchmark for evaluating LLMs in multi-agent negotiation scenarios [16]. Chatterjee et al. introduced `AgreeMate`, a framework for teaching LLMs strategic price haggling [17], focusing primarily on bargaining aspects rather than broader procedural requirements. Kim et al. developed the `MART` framework for active merchant NPCs capable of dynamic pricing [12]. While `MART` achieved progress in natural language trading, persistent challenges remain in maintaining consistent protocol adherence, preventing item hallucination, and ensuring accurate calculations.

Previous research has contributed valuable innovations across multiple dimensions of interactive system design. However, existing approaches typically focus on specific functionalities such as price negotiation [12] or emotion expression [14], rather than addressing comprehensive requirements of dynamic trading dialogues involving multiple rule-based states including inventory browsing, item selection, and multi-step confirmation procedures.

Such comprehensive trading systems require strict adherence to inventory constraints, pricing rules, and procedural protocols throughout entire dialogue flows while maintaining natural conversational patterns. A significant research gap exists in prompt engineering methodologies that enable LLMs to infer current trading states from dialogue context alone—essential for reliable dynamic trading interactions that maintain compliance with complex game system rules. While Dialogue State Tracking research [18] is crucial for this state inference, it often does not extend to controlling rule-governed dialogue flow.

The State-Inference-Based Prompting methodology introduced in this study systematically addresses these limitations through a unified approach to structured dialogue management. This methodology enables LLMs to autonomously infer trading states from dialogue context and maintain adherence to game system rules throughout complete trading processes without requiring external state management systems.

3 Proposed System

This section details the system architecture and core prompting methodology developed for natural language trading with game NPCs. The proposed State-Inference-Based Prompting approach guides LLMs to follow predefined dialogue states within goal-oriented interactions, with in-game trading serving as the primary application domain.

3.1 LLM-Driven Interaction Framework

The system enables players to naturally converse and trade with NPC merchants through a multi-stage process: (1) collect player input, (2) combine with game information including NPC inventory and dialogue history to construct contextual prompts, (3) call LLM API, (4) parse responses and update the game state, (5) present NPC dialogue to the player.

The most challenging aspect is the fundamental **unpredictability of dialogue**. Unlike menu-based trading in traditional games, natural language conversations allow players to change context dynamically at any time. Representative examples include:

- While an NPC is explaining item lists, suddenly asking *"How much is this?"* (immediate price inquiry)
- After getting price information, immediately proposing *"Can you make it 50 gold?"* (direct negotiation attempt)
- During active trade negotiations, suddenly asking *"Are there many monsters around lately?"* (casual conversation)
- Even abrupt *"Goodbye"* statements (conversation termination)

Since predicting player utterances is impossible, NPCs must **judge the current situation autonomously after listening to player input**. However, if the LLM misjudges the situation, inappropriate responses may occur, such as suddenly proceeding to payment during casual chat or failing to maintain transaction state consistency.

To achieve this capability, the prompt is structured to enable LLMs to accurately judge situations and respond according to game rules. Key components include:

- **System Instructions**: Guidelines defining the NPC's persona, role, and behavioral directives.
- **Game World Data**: Lists of all items available in the game and the NPC's inventory with item ID, name, quantity, and price.
- **Situational Context**: Information about the NPC's profile and current in-game environment such as current location and time.
- **Dialogue History**: A record of preceding dialogue turns between player and NPC, including the latest player input and inferred context from previous NPC turns.
- **Contextual and Trade Guidelines**: Rules and guidelines to infer states and identify when to transition between states (`<CONTEXT_GUIDELINES>` and `<TRADE_GUIDELINES>`; the full prompt is provided in Appendix A).
- **Expected Response Format**: Instructions for the LLM to structure its response in a format that facilitates system integration.

As guided by the Expected Response Format instructions, the LLM generates output as a single, well-formed JSON object. This approach relies on the LLM's capability to adhere to formatting instructions provided through prompting, rather than utilizing specialized structured output functionalities. The resultant JSON response facilitates seamless parsing and integration with game systems.

Key fields in the response structure include:

- **`context_reason`** (string): A summary or reasoning behind the LLM's understanding of the current context.
- **`context_type`** (string): The conversational context inferred by the LLM.
- **`context_details`** (object): Contains finer-grained context information, particularly for `TRADE` context. This includes the trade subcontext and an array of items pertinent to the current trade interaction.
- **`npc_dialogue`** (string): The natural language dialogue spoken by the NPC to the player.

This framework ensures that LLMs receive rich contextual information through detailed prompts and provide structured, actionable responses, enabling goal-oriented and rule-consistent NPC interactions that maintain both game world integrity and conversational naturalness.

3.2 State-Inference-Based Prompting

To address the dialogue unpredictability problem mentioned above, this work proposes **State-Inference-Based Prompting (SIBP)**. The core innovation of SIBP is enabling LLMs to determine *"what situation is happening right now?"* autonomously from the dialogue context alone.

The system defines three main situational contexts:

- **`NONE`** (general conversation): Casual interactions such as *"Hello," "Nice weather," "Monster news,"* etc.
- **`TRADE`** (trading conversation): Commercial interactions such as *"I want to buy a sword," "How much?" "Can you lower the price?"* etc.
- **`END_CONVERSATION`** (conversation ending): Termination phrases such as *"Goodbye," "I'll come back later,"* etc.

The LLM learns the characteristics of each situation from the `<CONTEXT_GUIDELINES>` section in the prompt, judges the appropriate situation based on player utterances, and then responds following rules appropriate for that situation.

For example, if a player suddenly asks *"Are there many monsters around lately?"* the LLM judges this as a `NONE` situation and responds with monster-related casual conversation instead of trade-related responses. This solves the *"sudden context changes"* problem mentioned in Section 3.1.

3.2.1 Trade Subcontexts. However, simply having one `TRADE` state is insufficient for implementing proper trading systems. For example, when a player says *"I want to buy a sword," "How much is that sword?"* and *"Yes, I'll buy it,"* the NPC's responses should be completely different for each case, requiring distinct behavioral guidelines.

If all these situations were handled with just a simple `TRADE` state:

- *"I want to buy a sword"* → Unclear whether NPC should immediately state the price or ask which sword
- *"How much?"* → Cannot determine which item's price is being asked or if negotiation is possible
- *"Yes, I'll buy it"* → It is unknown whether to proceed directly to payment or require additional confirmation

Therefore, the trading process must be decomposed more granularly to clearly define how NPCs should behave at each stage. For the scope of this study, focusing on player buying from an NPC, the following five trade subcontexts (i.e., sub-states) are defined:

- **`SHOW_INVENTORY`**: The state where the NPC displays or mentions items available for sale to the player.
- **`OFFER_SELL`**: The state where the NPC proposes selling specific items and states their price.
- **`NEGOTIATE_PRICE`**: The state where the player attempts to negotiate the price, and the NPC responds based on their character traits.
- **`CHECK_CONFIRMATION`**: A state where the NPC seeks final confirmation from the player before finalizing a purchase, designed to prevent abrupt transaction completion.
- **`CONFIRM_SELL`**: The state where the player confirms their purchase intent, leading to the final update of game data (e.g., deducting currency, removing items from inventory).

These states are not strictly sequential and can transition dynamically based on player input. For instance, a player might return to requesting other items while confirming a purchase, or switch between general conversation (`NONE`) and trade interaction (`TRADE`) to discuss the rarity or demand of specific items. This dynamic behavior requires an LLM approach capable of robust state inference.

3.2.2 Prompt Design Elements. But how can the system make LLMs properly distinguish and manage multiple trading states? Simply instructing *"respond appropriately to the situation"* leads to problems:

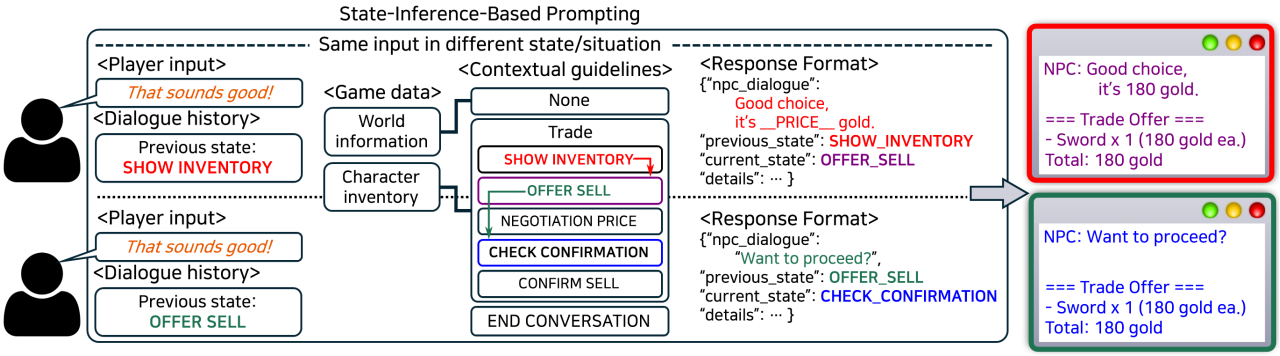


Figure 1: State-Inference-Based Prompting (SIBP) enables context-aware NPC responses. The same player utterance "That sounds good!" triggers different behaviors depending on the previous dialogue state: `SHOW_INVENTORY` → `OFFER_SELL` (price proposal) vs. `OFFER_SELL` → `CHECK_CONFIRMATION` (purchase confirmation). SIBP autonomously infers states from dialogue context while maintaining game system consistency.

- **State confusion**: When a player says *"Yes, that sounds good,"* the LLM cannot distinguish whether this is accepting a price offer or simply a positive reaction to an item description
- **State skipping**: Players saying *"I'll buy it"* immediately jumps to payment, skipping final confirmation steps
- **Inaccurate state inference**: LLMs failing to correctly identify the previous trading state from dialogue context alone, leading to inappropriate state transitions
- **Unclear reasoning**: Unable to understand why the LLM made certain state judgments, making debugging and improvement difficult

To solve these problems, SIBP incorporates four key design elements into the prompt within the `<TRADE_GUIDELINES>` sections. These elements are: (1) Basic State Definitions, (2) State Transition Conditions with Previous Context, (3) Directive to Identify Previous State, and (4) Directive to Respond including Previous State.

Element 1: Basic State Definitions. Provides foundational definitions for each trading state by establishing two aspects. First, it establishes the **context** of each state—what situation each state represents. For example, `SHOW_INVENTORY` is defined as *"the state where NPCs present available items to players,"* while `OFFER_SELL` is characterized as *"the state where NPCs propose specific items with pricing."* Second, it specifies the **behavioral rules** for each state—what actions should be taken within that state. This foundational definition ensures that LLMs understand both the situational context and appropriate behavioral guidelines for each state.

Element 2: State Transition Conditions with Previous Context. Solves both the **state confusion** and **state skipping** problems by defining the prerequisite conditions for entering each trading state, emphasizing the dependency on previous states. This element addresses state confusion by providing context-aware transition rules—for example, when a player says *"Yes, that sounds good,"* the appropriate response depends on whether the previous state involved item description (`SHOW_INVENTORY`) or price offering (`OFFER_SELL`). It prevents state skipping by establishing mandatory sequential flows. For instance, `CONFIRM_SELL` is accessible only when the previous state was `CHECK_CONFIRMATION` and the player provides positive confirmation.

Element 3: Directive to Identify Previous State. Addresses the **inaccurate state inference** problem by instructing the LLM to identify the previous trading state before determining the current response. The prompt includes commands such as *"Before responding, identify what the last trading state was based on the dialogue history."* This directive addresses the core challenge where LLMs may understand general conversation context but fail to accurately recognize the specific state identifiers from previous interactions.

Element 4: Directive to Respond including Previous State. Addresses both the **inaccurate state inference** problem and the **unclear reasoning** problem by requiring the LLM to document its identification of the previous trading state. While Element 3 focuses on INPUT processing (*"identify the previous state"*), Element 4 mandates OUTPUT transparency (*"document your state identification"*). The JSON response must include a `last_trade_context` field where the LLM records the identified previous state. This documentation enables system monitoring of the LLM's reasoning process, facilitates debugging, and provides a foundation for the improvement of state inference accuracy.

Figure 1 illustrates the core concept of SIBP through a concrete example. The figure demonstrates how the same player utterance *"That sounds good!"* triggers completely different NPC behaviors depending on the previous dialogue state. In the upper scenario, when the previous state was `SHOW_INVENTORY`, the LLM interprets the player's response as interest in an item and transitions to `OFFER_SELL`, providing a price proposal. In the lower scenario, when the previous state was `OFFER_SELL`, the same utterance is interpreted as price acceptance and transitions to `CHECK_CONFIRMATION`, seeking purchase confirmation. This context-aware state inference enables NPCs to maintain coherent dialogue flows while autonomously managing complex trading interactions without external state management systems.

3.3 Additional State-specific Rules

The expansion of SIBP by adding state-specific guidelines yields advantages in implementing game system functionalities and ensuring rule adherence. The system can achieve the following capabilities:

3.3.1 State-specific Data Referencing. This feature dynamically adjusts the data that NPCs reference based on dialogue context, addressing a challenge in maintaining both game world knowledge and transaction accuracy.

How it works: By defining item reference conditions based on states, the LLM responds differently:

- **`NONE` state (general conversation)**: References the complete game item list (`<GAME_ITEMS_LIST>`) for general world discussions
- **`TRADE` state (trading conversation)**: References only the NPC's inventory (`<CHARACTER_INVENTORY>`) for actual sellable items

Benefits: This enhances player immersion while ensuring accurate transaction capabilities. NPCs can naturally discuss various game world items during the general conversation but only propose actually sellable items during trading, preventing item hallucinations while maintaining conversational richness.

3.3.2 State-specific Post-processing. As highlighted in prior research [12], LLMs, being language models, can face challenges in precise mathematical computations. Even when correctly listing item quantities and individual prices, errors may occur in final price calculations, compromising transaction integrity.

Solution: SIBP implements accurate price calculations through state-specific post-processing. In the `OFFER_SELL` state—the initial price offering stage where price negotiation is not yet required—the system works as follows:

(1) **Placeholder usage**: The prompt includes the rule *"use __PRICE__ for the final price amount"*
(2) **Accurate calculation**: The system calculates the correct price based on responded item information and replaces the placeholder
(3) **Error prevention**: The accurately calculated price is included in dialogue history, ensuring correct price references in subsequent conversations

Response example:

- **Original LLM response**: *"Two iron swords and an adventurer's sleeping bag are __PRICE__ gold together."*
- **After system post-processing**: *"Two iron swords and an adventurer's sleeping bag are 150 gold together."*

Advantages: This approach achieves 99.7% accurate price calculation without additional computational overhead, eliminating the need for complex schemas or Tool use mechanisms. The single response processing maintains system efficiency while ensuring mathematical precision. This state-based prompting approach demonstrates how external system functionalities such as accurate calculation and post-processing can be integrated into LLM-driven interactions, providing a foundation for broader applications in rule-governed interactive systems.

4 Experiments and Results

This section presents comprehensive empirical evaluations of the State-Inference-Based Prompting (SIBP) methodology. The evaluation systematically assesses how effectively NPC LLMs adhere to predefined conversational flows during dynamic trading interactions and then demonstrates additional advantages of this approach through state-specific data referencing and post-processing.

4.1 Experimental Setting

All experiments employ conversations between a player-mimicking LLM (virtual player) and an NPC LLM to ensure controlled, reproducible evaluation conditions. One hundred independent trading dialogues were conducted using fixed seeds ranging from 0 to 99. In each dialogue, the virtual player initiated conversations with the NPC LLM. While the NPC LLM's prompt varied across different experimental conditions as detailed in subsequent subsections, the initial utterance from the virtual player remained consistent for each seed, establishing controlled starting points for interactions.

The conversations were structured around two systematically designed test scenarios:

(1) **Request for Specific Item Purchase:** Virtual players initiate conversations by requesting specific items. Items are randomly selected from sellable and unsellable categories up to six in total. Sellable items refer to items with sufficient stock in the NPC's inventory (20 items), while unsellable items exist in the game world but are absent from the NPC's inventory (an additional 32 items, totaling 52 distinct items in the game). Purchase quantities for each item are randomly selected from 1 to 5. For example: *"I'd like to purchase 3 basic iron swords and 5 mana potions."*
(2) **Request for Item Recommendation:** Virtual players initiate conversations mentioning purchase purposes such as *"Could you recommend me some items for goblin battle?"*

The virtual player prompts for each scenario are provided in Appendices B and C. Virtual players and NPCs engage in alternating dialogue turns. Conversation length is defined as the number of interaction rounds, where each round consists of one player turn followed by one NPC turn. Average conversation lengths were 5.17 rounds (SD=1.0, range: 3-8) for Scenario 1 and 6.05 rounds (SD=1.6, range: 2-12) for Scenario 2.

Unless specified otherwise in individual experiment descriptions, both virtual player LLMs and NPC LLMs utilized the `gemini-2.5-flash-preview-04-17` model with a temperature setting of 0.7 and a thinking budget of 0.

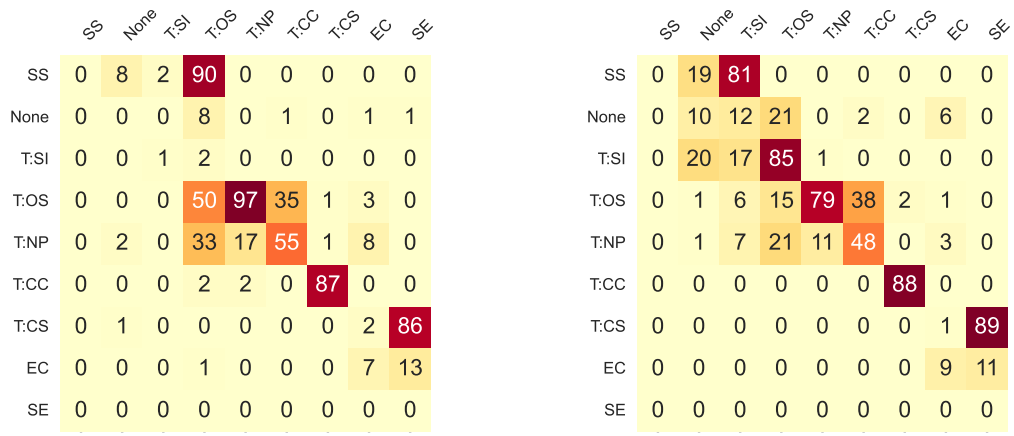
4.2 State Transition Compliance

A key advantage of SIBP lies in its ability to clearly define states and transitions in goal-oriented dialogues and systematically guide LLMs to follow them. SIBP achieves this through four core elements: (1) explanation of states, (2) explanation of state transition conditions, (3) directive to identify the last previous state, and (4) directive to include the last previous state in the response.

Experimental Design: To precisely measure the effect of each element, experiments systematically activated/deactivated the four elements to create multiple baselines. The complete SIBP includes all elements, while baselines activate only some elements. Table

Table 1: State Transition Compliance Rate (STCR) for different prompt element combinations. STCR represents the proportion of conversations that correctly followed the mandatory confirmation step before finalizing transactions, serving as a safeguard to prevent unintended purchases. In this table, the SIBP configuration also includes placeholder post-processing (PPP). Experiments in Section (4.4) will differentiate SIBP (not using PPP) from SIBP+PPP.

Scenario	Method	Prompt elements				STCR [%]
		(1) Explain States	(2) Explain Transitions	(3) Identify Prev. State	(4) Response Prev. State	
1	baseline1	O	X	X	X	79.55
	baseline2	O	O	X	X	84.09
	baseline3	O	O	O	X	77.27
	baseline4	O	O	X	O	94.32
	SIBP	O	O	O	O	97.73
2	SIBP	O	O	O	O	97.73



(a) Scenario 1: the conversation initiated by requesting to purchase specific items

(b) Scenario 2: the conversation initiated by asking for item recommendations based on a purchase purpose

Figure 2: State transition patterns in 100 dialogues. Cell values at row i and column j represent transitions from state i to state j. Abbreviations; SS(Session Start), T:SI(Show Inventory), T:OS(Offer Sell), T:NP(Negotiate Price), T:CC(Check Confirmation), T:CS(Confirm Sell), SE(Session End).

1 shows which elements are activated (*'O'*) or deactivated (*'X'*) for each method and their corresponding performance. This allows analysis of each element's individual contribution and the synergistic effects of their combinations.

Evaluation Metric: State Transition Compliance Rate (STCR) serves as the evaluation metric in this experiment. STCR is the proportion of conversations that correctly followed the "`CHECK_CONFIRMATION` → `CONFIRM_SELL`" sequence. To ensure a fair comparison across methods with varying numbers of `CONFIRM_SELL` state occurrences due to dialogue randomness, STCR was calculated based on the first 88 dialogues for each method where this state was reached. This count represents the minimum number of such occurrences observed across all methods within the 100 dialogues. Adherence to this specific state progression sequence was a critical rule intentionally enforced upon the LLM through the proposed prompt design. This sequence serves as an important procedural safeguard within game environments, designed to prevent unintended transactions by ensuring explicit player confirmation before purchase finalization.

Key Results: As shown in Table 1, SIBP with all elements achieved a compliance rate of over 97%. Element 4 proved impactful, improving the rate to 94% even without Element 3. Interestingly, while Element 3 did not by itself yield an improvement over simpler baselines (cf. baseline3 vs. baseline2), its combination with Element 4 was crucial for achieving the optimal compliance rate. This demonstrates that when LLMs identify previous states from dialogue history and explicitly include them in responses, adherence to designed dialogue flows is greatly enhanced.

State Transition Analysis: Figure 2 shows the observed state transitions as heatmaps. Each cell value matrix[i][j] represents the number of transitions from state i (previous) to state j (current). SS (Session Start) represents a conceptual point prior to the first

```
Player: Can I get five battle axes, two hunter's bows, five sturdy
pickaxes, and two rare gems?

NPC: Hmph. Five battle axes... and five sturdy pickaxes, is it? Sen
sible. The axes are quality steel. They're 180 gold apiece. Pickaxe
s are forged to last. Those will be 120 gold each. For that lot...
it's __PRICE__ gold. Hunter's bows? Rare gems? I don't stock them.
Not my specialty. You'll need to find another merchant for that sor
t of thing.
Context: TRADE
Context in Trade: OFFER_SELL
Items:
{'id': 'axe_01', 'name': 'Battle axe', 'quantity': 5, 'price': 180}
{'id': 'tool_02', 'name': 'Sturdy pickaxe', 'quantity': 5, 'price':
 120}

=== NPC Trade Offer ===
Trade Details:
  - Battle axe     x5 (180 gold each)
  - Sturdy pickaxe x5 (120 gold each)
Total price: 1500 gold
```

Figure 3: An example of an NPC LLM response distinguishing sellable items. The final output presented to the player consists of the NPC's post-processed dialogue and a structured trade offer list.

dialogue turn. SE (Session End) signifies the conceptual termination point occurring after the final NPC response.

Examining scenario-specific characteristics, Scenario 1 (direct specific item purchase) primarily begins with `OFFER_SELL` (Figure 2a), while Scenario 2 (recommendation-based purchase) mainly starts with `SHOW_INVENTORY` (Figure 2b). In both cases, valid state transitions were observed as designed, confirming that SIBP successfully induces natural dialogue flows appropriate to each situation.

4.3 State-specific Data Referencing

NPCs must balance natural conversation about the game world with accurate trading behavior. In general conversation, they can discuss all game items, but during trading, they should only propose items they actually sell, addressing the fundamental challenge of maintaining both world knowledge and transaction accuracy.

Experimental Design: This evaluation assesses SIBP's ability to reference different datasets based on dialogue state. The NPC prompt includes two datasets: `<GAME_ITEMS_LIST>` (IDs and names of all valid game items) and `<CHARACTER_INVENTORY>` (IDs, names, quantities, and prices of NPC's sellable items). The challenge is making the LLM select the appropriate dataset based on context.

Evaluation Metric: Sellable Item Response Rate (SIRR) serves as the evaluation metric. In this system, all NPC responses generated within the `TRADE` state are designed to populate an `items` field with relevant item data. SIRR is then calculated as the overall proportion of these `TRADE` state responses where the items field correctly contains only sellable item data. This calculation is aggregated across all 100 dialogues and provides a global measure of accuracy at the response level.

Key Results: The SIRR was 95.05% in Scenario 1 and 95.88% in Scenario 2 out of 485 and 534 NPC responses within the `TRADE` state, respectively. The most frequent error type involved the NPC assigning a price of 0 or a null value for unsellable items. In rare responses (0.4% and 0.2% in Scenario 1 and 2, respectively), the NPC failed to provide any item information when expected within a trade context. Overall, these high SIRR values, coupled with a clear understanding of the failure cases, demonstrate SIBP's effectiveness in state-specific data referencing.

4.4 State-specific Post-processing

LLMs struggle with precise arithmetic operations, making simple calculation errors like *"2×50 + 30 = 120 gold."* This poses significant problems for price calculations in trading, potentially undermining player trust and transaction integrity.

Experimental Design: Utilizing dialogues from Scenario 1, experiments compare 6 configurations (Table 2):

- **SIBP variants**: Basic SIBP, SIBP+PPP (with Placeholder Post-Processing), SIBP+SO (with Structured Output)
- **Model variants**: 2.5-flash (baseline), 2.0-flash (lighter version), 2.5-pro (high-performance)

Evaluation Metric: The evaluation measures price accuracy, completion tokens, thought tokens, and response time in `OFFER_SELL` state and other price-related states. Price accuracy is the percentage of NPC responses where the total price stated by the LLM matches the sum calculated from the item details (i.e., prices and quantities) also provided within the same LLM response.

Key Results: SIBP+PPP showed consistent improvements in accuracy while maintaining computational efficiency across multiple evaluation dimensions. First, in terms of accuracy, SIBP+PPP reached 100.0% in the `OFFER_SELL` state, compared to 79.7% with basic SIBP. This improvement also extended to subsequent states, where SIBP+PPP achieved 99.7% accuracy versus 82.3% for basic SIBP. These results suggest that accurate initial pricing may help reduce downstream errors in multi-turn dialogues.

Second, these accuracy gains were achieved without notable increases in computational cost. Token usage (398.6 vs 394.8) and response time (2.3 seconds) remained comparable to basic SIBP, while thought token usage was held at zero. By contrast, SIBP+SO introduced additional complexity, requiring 371.8 thought tokens and a longer response time of 4.1 seconds. Furthermore, the high variability in thought tokens observed in other configurations (e.g., 553.7 for SIBP+SO, 1013.6 for 2.5-pro) reflects the differing internal reasoning needs across dialogue types, with simpler exchanges requiring fewer resources and more complex negotiations involving significantly more.

Third, SIBP+PPP also showed promising results with smaller models. When tested with 2.0-flash, it achieved 100.0% accuracy in `OFFER_SELL` and 91.8% in other states, outperforming the basic SIBP setup, which achieved 49.4% and 67.3%, respectively. This indicates the method's potential for generalization across model variants.

Lastly, SIBP+PPP using 2.5-flash reached the same level of accuracy as Basic SIBP with the 2.5-pro (100.0%) while requiring significantly fewer resources: 0 vs 2398.5 thought tokens and 2.3 vs 27.7 seconds in response time. These findings suggest that incorporating state-specific post-processing may allow smaller models to approximate the accuracy of larger ones, with considerably lower computational demands.

5 Discussion

5.1 Hallucination Issues

The LLM-driven interaction framework developed in this work requires strict consistency with the underlying game system. Even small deviations, such as generating `SHOW_INVENTOR` instead of

Table 2: Performance comparison across six system configurations. Data is presented for the `OFFER_SELL` state and for Others (a collective of `NEGOTIATE_PRICE`, `CHECK_CONFIRMATION`, and `CONFIRM_SELL` states). The X/Y notation in each column header indicates the total number of NPC responses within the `OFFER_SELL` state (X) and Others states (Y), respectively, over 100 dialogues. Price accuracy is a percentage; Token usage and response time values are mean (standard deviation).

Metric	State	SIBP 2.5-flash (192/283)	SIBP+SO 2.5-flash (209/294)	SIBP+PPP 2.5-flash (186/296)	SIBP 2.0-flash (174/284)	SIBP+PPP 2.0-flash (194/281)	SIBP 2.5-pro (211/288)
Price accuracy	OFFER_SELL	79.7	82.3	100.0	49.4	100.0	100.0
	Others	82.3	88.4	99.7	67.3	91.8	100.0
Completion tokens	OFFER_SELL	394.8 (71.3)	375.4 (67.1)	398.6 (62.5)	448.5 (82.6)	453.0 (84.7)	422.0 (65.1)
	Others	377.4 (71.0)	347.2 (64.8)	378.4 (61.0)	393.3 (67.9)	396.4 (71.2)	382.3 (48.8)
Thoughts tokens	OFFER_SELL	0 (0)	371.8 (553.7)	0 (0)	0 (0)	0 (0)	2398.5 (1013.6)
	Others	0 (0)	215.3 (366.6)	0 (0)	0 (0)	0 (0)	1715.0 (999.2)
Response time	OFFER_SELL	2.3 (0.3)	4.1 (2.5)	2.3 (0.3)	2.8 (0.6)	2.8 (0.5)	27.7 (10.4)
	Others	2.2 (0.4)	3.2 (1.8)	2.2 (0.3)	2.5 (0.7)	2.5 (0.4)	21.0 (10.5)

`SHOW_INVENTORY`, or referencing an item as `sleeping_bag_01` instead of the correct `sleeping_bag`, can lead to failures in state transitions or item retrieval, compromising system integrity.

State Name and Item Information: No hallucinations were observed in the generation of state names or item attributes during the experiments. This reliability is attributed to deliberate prompt design. State names were clearly defined and limited to a set of eight valid options, making accurate selection easier for the model. Item information, including ID, name, quantity, and price, was represented as a JSON string, which effectively constrained the LLM's outputs.

Placeholder Keywords: Placeholder keywords used for post-processing require careful selection. The concise keyword `__PRICE__` consistently produced accurate and recoverable outputs. In contrast, more complex variants such as `__PRICE_PLACEHOLDER__` occasionally resulted in malformed completions like `__PRICE_PLACEHOLDER_` or `__PRICE_PLACEHOLDE__`. These inconsistencies highlight the importance of using short, unambiguous keywords to minimize generation errors and ensure reliable post-processing.

5.2 Limitation and Future Works

Scalability and Generalizability: The current experiments were conducted within a defined set of trading states and rules. Future work should rigorously evaluate the scalability of this approach across significantly larger numbers of conversational states, more intricate and interdependent game rules, and substantially larger item inventories that might challenge context window limits or retrieval accuracy. Additionally, testing with longer, more nuanced dialogue histories will be crucial for understanding the performance envelope of this prompting technique under increased complexity.

LLM Model Diversity: The current implementation utilized a specific LLM architecture. Investigating the adaptability and performance of this prompting methodology across a range of different LLMs is a critical next step. This should include not only other large-scale models but also an exploration of effectiveness with smaller language models (SLMs). If comparable adherence to complex rule sets can be achieved with SLMs, it would significantly enhance the feasibility of deploying sophisticated NPCs in resource-constrained game environments, potentially reducing both latency and computational costs.

Broader Applications Beyond Gaming: The core principles of State-Inference-Based Prompting have significant potential beyond the realm of game NPCs. Potential applications include creating more natural and capable interactive kiosks for retail or information services, where the system could seamlessly transition between general queries, product information, and transaction processing. Other potential areas include advanced customer support chatbots that can follow complex troubleshooting or sales workflows, and educational tools that guide users through learning modules in a state-aware manner.

6 Conclusion

This work introduces State-Inference-Based Prompting (SIBP), a comprehensive methodology that enables reliable LLM-driven trading through autonomous dialogue state inference and systematic rule adherence. By decomposing trading interactions into five distinct states within a unified prompt framework, SIBP addresses critical challenges that have hindered the deployment of LLM-based NPCs in commercial games: item hallucinations, inventory inconsistencies, and calculation errors that erode player trust.

Empirical evaluation demonstrates SIBP's effectiveness across multiple dimensions. The methodology achieves >97% state transition compliance, >95% item referencing accuracy, and 99.7% price calculation precision, eliminating major barriers to commercial deployment. These results establish SIBP as a practical solution that bridges the gap between LLM capabilities and game system requirements while maintaining computational efficiency.

The broader implications extend beyond gaming. SIBP's structured approach to prompt engineering provides fundamental advances for rule-governed interactive systems where reliability and consistency are paramount. This methodology demonstrates how sophisticated state management can transform LLM capabilities into trustworthy, task-oriented systems, establishing a foundation for the next generation of intelligent interactive agents that combine natural language flexibility with structured system reliability.

References

[1] Junyu Luo et al. A survey on large language model based autonomous agents. *arXiv preprint arXiv:2503.21460*, 2024.

[2] Mahdi Farrokhi Maleki and Richard Zhao. Procedural content generation in games: A survey with insights on emerging llm integration. *arXiv preprint arXiv:2410.15644*, 2024.

[3] Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. *arXiv preprint arXiv:2304.03442v2*, Aug 2023.

[4] Li Song. LLM-Driven NPCs: Cross-Platform Dialogue System for Games and Social Platforms. *arXiv preprint arXiv:2504.13928*, 2025.

[5] Xuefeng Li et al. Character generalization: A study on out-of-domain role-playing dialogue agents. *arXiv preprint arXiv:2501.15427*, 2025.

[6] Tereza Todova and Peter Bruza. A quest for information: Enhancing game-based learning with LLM-Driven NPCs. In *Proceedings of CESCG 2025: The 29th Central European Seminar on Computer Graphics*, 2025.

[7] K. I. Gero et al. Procedural Artificial Narrative using Generative AI (PANGeA). In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE)*, 2024.

[8] Chen Wang et al. CivAgent: A Human-like Agent for Complex Strategy Games. *arXiv preprint arXiv:2502.20807*, 2025.

[9] Relu Games. Magical Mike Senpai, Here My Spell Is. `https://store.steampowered.com/app/2696400`, 2025.

[10] Relu Games. Uncover The Smoking Gun. `https://store.steampowered.com/app/2492290`, 2025.

[11] Jaewoong Cho. SmartJoy in inZOI: LLM-based Live Simulation Engine. Presentation at Game Developers Conference (GDC) 2025. Information based on report from Eneba Hub News [3], 2025.

[12] Byungjun Kim, Minju Kim, Dayeon Seo, and Bugeun Kim. Leveraging large language models for active merchant non-player characters. *arXiv preprint arXiv:2412.11189v2*, Dec 2024.

[13] Zhi Yuan et al. RPGBench: A Comprehensive Benchmark for Evaluating Large Language Models as Role-Playing Game Engines. *arXiv preprint arXiv:2502.00595*, 2024.

[14] Alessandro Marincioni, Myriana Miltiadous, Katerina Zacharia, Rick Heemskerk, Georgios Doukeris, Giulio Barbero, and Mike Preuss. The effect of llm-based npc emotional states on player emotions: An analysis of interactive game play. In *2024 IEEE Conference on Games (CoG)*, pages 1–6, 2024.

[15] Frederik Roland Christiansen, Linus Nørgaard Hollensberg, Niko Bach Jensen, Kristian Julsgaard, Kristian Nyborg Jespersen, and Ivan Nikolov. Exploring presence in interactions with llm-driven npcs: A comparative study of speech recognition and dialogue options. In *30th ACM Symposium on Virtual Reality Software and Technology (VRST '24)*, pages 1–11, Trier, Germany, Oct 2024.

[16] Shahin Abdelnabi et al. LLM-Stakeholders Interactive Negotiation: A Benchmark for Cooperation, Competition, and Maliciousness. *arXiv preprint [6, 7]*, 2024.

[17] Ainesh Chatterjee, Samuel Miller, and Nithin Parepally. AgreeMate: Teaching LLMs to Haggle. *arXiv preprint arXiv:2412.18690*, 2024.

[18] Yujie Feng, Zexin Lu, Bo Liu, Li-Ming Zhan, and Xiao-Ming Wu. Instruction-Tuning for Dialogue State Tracking. *OpenReview [4]*, 2023.