International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Engineering Reliable Agentic AI: Decision-Making in Real-World Deployments

Rimjhim (Rémy) Singh

Founder and Founding AI Engineer, Rikanverse Ltd., London, United Kingdom

Abstract

Large language model (LLM) agents are increasingly deployed to give everyday guidance whose correctness depends on rules that users do not know and on personal facts that the agent must first elicit. At each turn, such an agent must decide whether to ask a clarifying question, act through a tool or answer, and it must do so reliably. This paper presents Rikanverse®, a Proactive, Multilingual, Voice-Native Agentic AI Application, built on a Reasoning-and-Acting (ReAct) Orchestration Engine and designed, built and deployed end-to-end solo by the Author. Rikanverse® serves both sides of the market: tenants describe their needs by voice or text and receive matched listings with practical guidance, while landlords register listings by speaking instead of completing forms. The agent treats clarification as an explicit action, acts through a small, typed tool set, maintains persistent session state, and checks every draft answer before release so that only reliable answers reach the user. The paper describes the system, reports preliminary user evidence from a survey of 30 respondents, and proposes an evaluation protocol in which answer reliability serves as the central measure.

Keywords: Artificial Intelligence, Human-AI Interaction, LLM Agents, Voice Interfaces

1. Introduction

Many everyday decisions depend on rules that ordinary users cannot easily interpret. Renting a home is a clear example. Whether a tenant owes Council Tax, which tenancy agreement applies or whether a deposit is protected depends on the person's situation, such as student status, visa or tenancy type, rather than on the text of a listing. The author encountered this problem directly. After her MSc ended while her student visa remained valid, no property portal gave a clear answer on Council Tax liability. A former Council Tax officer eventually explained that liability follows the course end date, not the visa.

This domain is a useful testbed for agent design for three reasons. Correct answers often require facts that the agent must first elicit from the user; the same question can have different answers for different users; and a fluent but incorrect answer on tax or tenancy matters can cause financial or legal harm. This motivates the central research question: how should an agent decide between asking, acting and answering so that its guidance remains reliable when errors are costly and information is incomplete? To study this question, the author built Rikanverse®, which is currently deployed in closed beta. The paper makes three contributions: (1) the design of a deployed agent that treats asking, acting and answering as explicit choices; (2) a response-checking step between the agent's draft and the user that releases only answers that pass reliability checks; and (3) an evaluation protocol that measures answer reliability.

2. Related Work

ReAct (Reasoning and Acting) interleaves reasoning traces with actions, allowing a model to gather

information and revise its plan during a task (Shunyu Yao, et al., 2023) [1]. Toolformer shows that language models can learn when to call external tools (Timo Schick, et al., 2023) [2]. Work on clarification questions treats asking as a decision with measurable value (Sudha Rao, et al., 2018) [3], and selective prediction formalises when a model should withhold a prediction it is unsure of (Yonatan Geifman, et al., 2017) [4]. Research on indirect prompt injection shows that instructions hidden in content an agent processes can redirect its behaviour, a risk that grows once an agent can create records or take actions (Kai Greshake, et al., 2023) [5]. This work focuses on a less-studied question: how an agent should decide between asking, acting and answering so that the answers it gives stay reliable. It is an applied systems study rather than a new learning method.

3. Materials and Methods

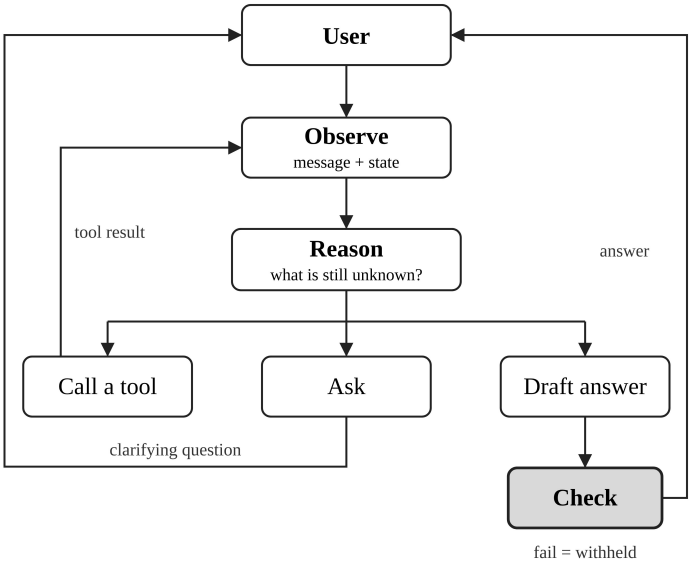
3.1 Rikanverse®

Rikanverse® is deployed in closed beta as a Proactive, Multilingual, Voice-Native Agentic Artificial Intelligence (AI) App serving both sides of the United Kingdom (UK) property market. Tenants and buyers describe the home they want, including lifestyle preferences, by voice or text; ask practical questions, for example about Council Tax, tenancy agreements or mortgages; and are matched to listings on Rikanverse. Landlords and agents register a property by speaking or typing instead of completing forms and can ask market questions such as what rent to charge. Users can also converse in their native language.

3.2 Decision Loop

At each turn, Rikanverse® observes the user's message and its session state, reasons about what is still unknown, and takes one of three actions: ask, call a tool or draft an answer (Figure 1). A draft is never sent directly to the user; it must first pass the response checks described in Section 3.5.

Figure 1: Decision Loop of Rikanverse



In Figure 1, clarifying questions go directly to the user, tool results update the agent's state, and draft answers are released only after passing the response checks.

3.3 Design Elements

- **Action space.** The agent acts through a small set of typed tools covering listing search, practical guidance, structured listing creation and hand-off to external platforms. Keeping the action space small limits what a faulty plan can do.

- **Personas and state.** One reasoning engine serves tenants and landlords as separate personas, each with its own memory and permissions. Session state persists across turns, so earlier constraints remain available in long conversations.
- **Modalities.** Voice and text run as separate pipelines with their own latency budgets, because every check adds delay and voice users notice delay sooner.
- **Proactivity.** Rather than waiting for complete input, the agent guides the conversation, anticipating missing details and asking for them, much as an experienced letting agent would.

3.4 Why Rikanverse® Is Not a Chatbot

A conventional chatbot maps each user message to a reply, either through a scripted flow or a single language-model call, and the assistants on most property portals match words in the query against listing text. Rikanverse® also relies on foundation models, but its behaviour is determined by the orchestration around the model rather than by the model alone. It differs from a chatbot in seven technical respects (Table 1).

Table 1: Conventional Chatbot Compared with the Rikanverse Agent

Dimension	Conventional chatbot	Rikanverse agent
Control flow	One message, one reply	Multi-step reason-and-act loop within each turn
Missing information	Guesses or replies "not found"	Asks a targeted clarifying question
Actions	Produces text only	Calls typed tools: search, listing creation, hand-off
Memory	Current message or a short window	Persistent session state across the conversation
Users	A single role	Tenants and landlords on one engine, with separate state and permissions
Uncertain answers	Released as fluent text	Checked for reliability before release
Voice	Text-to-speech added to a text bot	Separate voice pipeline with its own latency budget

The difference is clearest in a concrete case. An international student asks: "I don't have a UK credit history or a guarantor; can I get this flat?" A keyword-based assistant searches the advert for "credit history" or "guarantor". Because adverts rarely mention referencing edge cases, it finds nothing and the conversation ends, even though the flat may be available to the student. Rikanverse instead works through the situation as a short reasoning-and-acting trace:

- **Reason.** No UK credit history and no guarantor means a standard reference check is likely to fail, but alternatives may exist.
- **Act.** Look up which referencing alternatives the letting agency accepts for this listing.
- **Observe.** The agency accepts several months' rent in advance or an approved guarantor service.
- **Respond.** Explain both routes and offer to start the application with either one.

Three properties make this an agent rather than a chatbot: the system decides its own next step, it acts on the world through tools rather than only producing text, and it releases an answer only after checking that it is reliable. The model supplies language and reasoning; the loop, the action space, the state and the checks determine what the system actually does.

3.5 Response Checking

The design principle is reliability: the agent should not guess. Answers on regulated matters, such as Council Tax and tenancy rules, are designed to be grounded in information drawn from verified official sources. Before a draft answer is released, it passes a series of checks. If a draft fails, the agent asks for the missing information or, where it still cannot give a reliable answer, directs the user to the relevant official advisory service.

The checks address four concerns: whether an answer is supported, whether it is consistent with the user's stated situation, whether the agent is sufficiently confident, and whether the input and output are safe with respect to prompt injection and personal data. Implementation details are withheld because the system is under active commercial development. These checks reduce risk but provide no formal guarantee, and checks that themselves rely on language models can err.

3.6 Evaluation Protocol

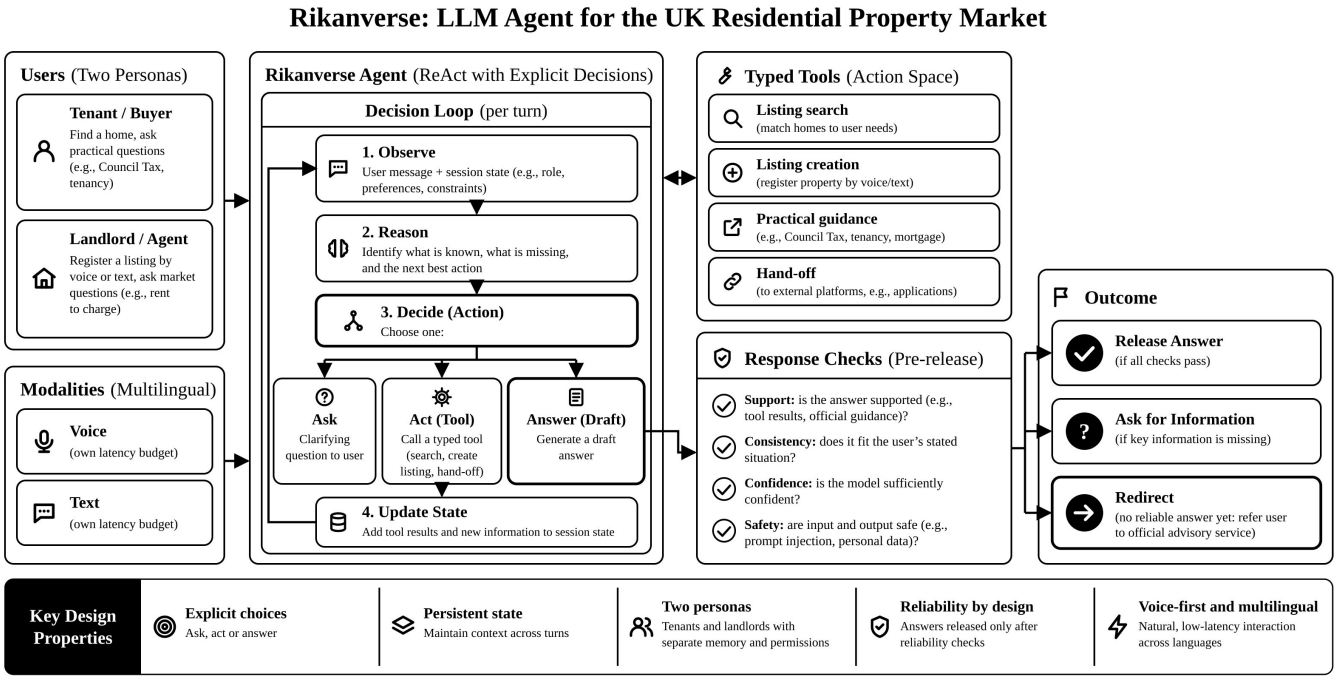


Figure 2: Overview of the Rikanverse Agent Architecture

A test set of tenancy and Council Tax questions, each with a reference answer reviewed by a domain expert, will be used to measure reliability. Each answer is scored as correct or wrong, and the wrong rate serves as the main safety measure. Because an agent could avoid errors simply by answering less, the protocol also records how often the agent redirects the user instead of answering. Three further experiments follow:

- **Ablation.** Remove one check at a time and measure the change in wrong answers.
- **Multilingual consistency.** Pose the same questions in several languages and compare correct and wrong rates.
- **Robustness and latency.** Measure the injection attack success rate with and without filtering, and voice round-trip latency for each configuration.

An overview of the complete system, from users and modalities through the decision loop, typed tools and response checks to the final outcome, is shown in Figure 2.

4. Results and Discussion

4.1 Preliminary User Evidence

The system is deployed in closed beta and works end-to-end in recorded sessions. Two online surveys, conducted in May and September 2026 with 30 respondents of whom 18 were tenants, measured perceived demand for such a system (Table 2).

Table 2: Share of Survey Respondents Agreeing with Each Statement

Statement	All (N = 30)	Tenants (N = 18)
Filter-based search fails my questions	60.0%	66.7%
Would value an always-available AI guide	90.0%	88.9%
Would try a voice-first agent	90.0%	94.4%

Source: Author's online surveys, May and September 2026.

Of the 27 respondents who valued an AI guide, 14 made it conditional on the information being accurate. Users therefore treat accuracy, not convenience, as the main requirement, which supports the protocol's focus on answer reliability.

4.2 Design Lessons

Three design lessons emerge. First, asking is an action: many questions cannot be answered without one more fact, so the clarifying question belongs in the action space rather than in the prompt. Second, in a domain where errors are costly, a reliable answer matters more than a fast or fluent one. Third, safety and latency must be designed together, because every check costs time and voice exposes that cost immediately.

5. Conclusion

This paper presents a deployed agent that treats asking, acting and answering as explicit choices and releases an answer only after it passes a series of reliability checks. The contribution is a working system together with an evaluation design centred on answer reliability. The next step is to run that evaluation and report correct and wrong rates, with and without each check.

References

1. Shunyu Y., Jeffrey Z., Dian Y., Nan D., Izhak S., Karthik N., Yuan C., "ReAct: Synergizing Reasoning and Acting in Language Models", Proceedings of the International Conference on Learning Representations (ICLR), 2023. https://arxiv.org/abs/2210.03629
2. Timo S., Jane D., Roberto D., Roberta R., Maria L., Luke Z., Nicola C., Thomas S., "Toolformer: Language Models Can Teach Themselves to Use Tools", Advances in Neural Information Processing Systems (NeurIPS), 2023, 36. https://arxiv.org/abs/2302.04761
3. Sudha R., Hal D., "Learning to Ask Good Questions: Ranking Clarification Questions Using Neural Expected Value of Perfect Information", Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL), 2018. https://arxiv.org/abs/1805.04655
4. Yonatan G., Ran E., "Selective Classification for Deep Neural Networks", Advances in Neural Information Processing Systems (NeurIPS), 2017, 30. https://arxiv.org/abs/1705.08500
5. Kai G., Sahar A., Shailesh M., Christoph E., Thorsten H., Mario F., "Not What You've Signed Up For: Compromising Real-World LLM-Integrated Applications with Indirect Prompt Injection", Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security (AISec), 2023. https://arxiv.org/abs/2302.12173