

End-user automation authoring in VR and AR through conversations: patterns, errors and user recovery strategies

Alessandro Carcangiu, Marco Manca, Jacopo Mereu, Carmen Santoro, Ludovica Simeoli & Lucio Davide Spano

To cite this article: Alessandro Carcangiu, Marco Manca, Jacopo Mereu, Carmen Santoro, Ludovica Simeoli & Lucio Davide Spano (28 Aug 2026): End-user automation authoring in VR and AR through conversations: patterns, errors and user recovery strategies, Behaviour & Information Technology, DOI: [10.1080/0144929X.2026.2714168](https://doi.org/10.1080/0144929X.2026.2714168)

To link to this article: <https://doi.org/10.1080/0144929X.2026.2714168>



© 2026 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group



Published online: 28 Aug 2026.



Submit your article to this journal [↗](#)



Article views: 291



View related articles [↗](#)



View Crossmark data [↗](#)

End-user automation authoring in VR and AR through conversations: patterns, errors and user recovery strategies

Alessandro Carcangiu ^a, Marco Manca ^b, Jacopo Mereu ^a, Carmen Santoro ^b, Ludovica Simeoli ^b and Lucio Davide Spano ^a

^aDepartment of Mathematics and Computer Science, University of Cagliari, Cagliari, Italy; ^bCNR-ISTI, Pisa, Italy

ABSTRACT

Automations defined for XR allow end-users to customise environments, but current menu-based interfaces pose challenges for end-user developers. Chatbots powered by Large Language Models (LLMs) could simplify the automation creation process, yet how users articulate their intentions remains underexplored. This work examines how end-users express event-condition-action (ECA) automations in both a Virtual Reality (VR) museum and an Augmented Reality (AR) smart home using a consistent set of tasks. Initially, we used a Wizard of Oz technique in a Wizard of Oz study to determine how users would communicate with a hypothetical agent, identifying dialogue patterns. Building on this, we tested a prototype LLM-based chatbot, revealing that conversational automation authoring exhibits a consistent, high-level dialogue structure across XR settings. Interactions with the LLM lead to more goal-oriented behaviours compared to dialogues with human agents. An analysis of errors shows that system errors were more likely to lead to cascading effects while users showed limited awareness of errors. These findings inform the design of chatbot support for expressing automations in XR environments.

ARTICLE HISTORY

Received 31 December 2025
Accepted 27 July 2026

KEYWORDS

eXtended reality; end-user development; immersive authoring; large language models; dialogue patterns; recovery strategies

1. Introduction

Within End-User Development (EUD), rule-based approaches allow users to shape and adapt system behaviour by defining configurable rules. This paradigm supports the customisation of dynamic environments in situations where requirements frequently change or where users need fine-grained control over automation (Paternò and Santoro 2019). Such approaches have been widely adopted in home automation contexts (Corno, De Russis, and Monge Roffarello 2019; Desolda, Ardito, and Matera 2017; Ghiani et al. 2017; Ur et al. 2014; Z. Inc 2022), where end users can specify automation rules through user-friendly interfaces, without relying on traditional programming skills. By abstracting technical complexity, these systems enable users to control and coordinate smart devices—such as lighting or security components—through rule-based automation mechanisms.

More recently, similar rule-based paradigms have been explored in the authoring of immersive extended reality (XR) experiences (Artizzu et al. 2022; Blečić et al. 2021; Fanni et al. 2019; Frau et al. 2023; Yigitbas et al. 2023; Zarraonandia et al. 2016). In these scenarios, rules are used to define interactions and dynamic

behaviours within virtual environments, enabling users to construct experiences aligned with their goals and preferences. Existing authoring tools typically require users to define their automation intent by explicitly specifying events, conditions, and actions through dedicated interfaces. These rules are commonly expressed as Event–Condition–Action (ECA) structures, where an event triggers the evaluation of conditions that, if satisfied, lead to the execution of an action. The translation process from users' intent to ECA rules requires a level of abstraction that may pose a significant barrier, particularly for users with limited technical expertise.

While most current systems rely on visual programming metaphors or constrained linguistic patterns (e.g. *if this, then that* Ur et al. 2014), recent advances in natural language processing, especially those driven by Large Language Models (LLMs), open up new opportunities for supporting rule creation. Conversational agents powered by LLMs can enable users to express automation logic through conversations in natural language, potentially lowering the cognitive and technical barriers associated with rule definition. In such interactions, the

agent is expected to help users articulate their intent, progressively refine it, and translate it into executable rules that reflect their expectations. Initial research has begun to investigate the use of LLM-based conversational systems to support the creation and modification of automations in different application domains (Carcangiu et al. 2025, 2026; Gallo, Paternò, and Malizia 2024). However, there is still limited understanding of how these conversations are structured and which interaction patterns emerge when users define automation rules through dialogue.

To systematically characterise conversational automation authoring in XR, we address the following research questions:

- (1) RQ1: What is the structure of conversations between end-users and an intelligent agent assisting them in authoring automations in natural language for XR environments?
- (2) RQ2: What recurring dialogue patterns emerge during conversational automation authoring?
- (3) RQ3: What types of errors occur in conversational automation authoring, and how do users detect and recover from them?

To address these questions, this paper extends the findings reported in Carcangiu et al. (2025) by investigating the structure of user-agent dialogues and the recurring patterns that arise when end users create automation rules via conversational interaction. Our goal is to characterise how conversational interaction shapes user strategies, dialogue structures, and error recovery in XR automation authoring. As a first step, we analysed data collected from a Wizard-of-Oz (WoZ) study, in which a researcher simulated the behaviour of a conversational agent across two application scenarios: a virtual reality museum and an augmented reality smart home. Both case studies are drawn from consolidated application areas of immersive technology: mobile AR for smart home interaction (Ariano et al. 2023; Heun, Hobin, and Maes 2013; Mattioli and Paternò 2025) and VR for museums (Bekele et al. 2018; YiFei and Othman 2024). Also, they were selected because they represent two complementary types of XR applications differing along several dimensions, e.g. device modality (head-mounted vs. hand-held), environment type (public/cultural vs. private/domestic), and task type (exploratory vs. goal-oriented).

Leveraging on the results of this WoZ study, we repeated the experiment using a working chatbot prototype called Tell-XR (Carcangiu et al. 2026).¹ We created a working VR museum and an AR smart home environments, and repeated similar tasks to assess the similarities and differences in dialogue structure and

patterns when using an LLM-based agent. Our main findings are that conversational rule creation follows a stable, high-level dialogue structure that may generalise across XR settings and application domains. Such a structure includes (i) the definition of the overall automation goal, (ii) the exploration of the possible objects and services to achieve it, (iii) the refinement of the overall plan with detailed options and parameters, (iv) the confirmation of the automation rule and (v) the export of the rule into the system for execution. We discovered several patterns that characterise different approaches to defining an automation. Finally, we found that interactions with the LLM-based agent encourage more guided, goal-oriented behaviour compared to exploratory dialogues.

In addition, to better understand the dynamics of the interaction between users and the conversational agent, we analysed the errors that appear during such dialogues, to enable the identification of critical breakdowns that can occur when using conversational agents to support unskilled users in creating automations in XR settings. We systematically identified different characteristics of the errors, including their types, frequencies, dependencies, and associated recovery strategies that users employ to repair them. We found that task complexity is a major factor influencing errors during rule creation and that system-generated errors propagate and affect task completion, especially in the VR setting. Additionally, in both settings, users tended to show limited error awareness and potential over-reliance on the system's accuracy. This analysis provided useful insights not only into users' awareness/mental models of the overall system, in terms of its objects and capabilities, but also into users' expectations and attitudes towards the accuracy of the conversational agent.

We organised the paper as follows. After discussing related work (Section 2), we describe and analyse the dialogues we initially collected in the WoZ study (Section 3). Then, we briefly describe the features and implementation of the chatbot prototype, along with the procedure and the results of the user study employing it (Section 4). Next, we discuss the relevance and the implications of our findings (Section 5). Finally, we conclude the paper and identify directions for future work (Section 6).

2. Related work

This section summarises existing work in the End-User Development (EUD) literature that is relevant to the topics addressed in this paper. Appendix 1 presents four tables that analytically compare our research with the cited approaches in the literature.

2.1. Rule-based approaches in EUD

Rule-based customisation has been extensively investigated as a means to lower the entry barriers for end users. A widely adopted paradigm in this context is Trigger-Action Programming (TAP), popularised by platforms such as IFTTT (IFTTT n.d.). TAP allows end-users to define automation rules that connect a trigger – or a logical combination of multiple triggers – to one or more actions, following the schema ‘when trigger, do action’.

However, when many devices are involved, TAP can cause user confusion and lead to reasoning errors (Ariano et al. 2023; Brackenbury et al. 2019; J. Huang and Cakmak 2015). TAP is widely used in IoT environments, both in research prototypes (Andrao et al. 2023; Bellucci et al. 2019; Fogli, Peroni, and Stefini 2017; Ikeda and Szafr 2024; Leonardi et al. 2019; Monge Roffarello and De Russis 2023; Stefanidi et al. 2019, 2020) and in commercial platforms (IFTTT n.d.; Z. Inc 2022), and variants incorporating if-then-else constructs have also been explored (Coutaz and Crowley 2016; Dey and Newberger 2009; Kubitza and Schmidt 2015).

A known limitation of TAP, also noted by J. Huang and Cakmak (2015), is that it does not explicitly distinguish between events (instantaneous signals) and states (boolean conditions), conflating the two into a generic trigger. The Event-Condition-Action (ECA) paradigm resolves this ambiguity through an explicit condition clause, namely a guard that, upon the occurrence of an event, is evaluated to decide whether the action should be executed. More specifically, once an event is detected, the system evaluates the associated condition(s) and performs the action only if they are satisfied (conditions can also be combined through logical operators such as OR or AND). Although ECA can support multiple conditions, many languages designed for end users restrict this expressiveness to simplify rule specification (Coutaz and Crowley 2016). ECA rules have been employed in several EUD application areas, including wellness monitoring systems (Barricelli et al. 2019) and tools aimed at supporting the debugging of TAP rules (Manca et al. 2019). Artizzu et al. (2022) adopt ECA rules expressed in natural language to define behaviours in virtual reality environments. The ECA paradigm has also been applied to home automation (Ariano et al. 2023; De Russis and Corno 2015; Desolda et al. 2021; Zhao et al. 2021) and to point-and-click game development (Blečić et al. 2021; Fanni et al. 2019).

While TAP and ECA provide effective abstractions for end-user rule specification, they generally rely on structured interfaces, which might not be very intuitive

for unskilled users. To make rule authoring more accessible, a growing body of work has investigated natural language as an input modality for rule formulation, especially in the IoT domain. For example, Instructable-Crowd (T. H. K. Huang et al. 2019) and HeyTAP (Corno, De Russis, and Monge Roffarello 2020) focus on translating user needs into executable IF-THEN rules. Noura, Heil, and Gaedke (2020) analysed how users naturally express their automation goals, observing that, in contrast to the strict command-based interaction adopted by systems such as IFTTT, users frequently rely on indirect expressions (e.g. ‘it’s too cold here’). More recently, Gallo, Paternò, and Malizia (2024) presented RuleBot++, a GPT-4-powered conversational agent that supports non-expert users in creating and modifying trigger-action automations for IoT and smart home devices using natural language. The authors also report a limited analysis of the errors that may arise during chatbot-user interactions; however, the study focuses exclusively on IoT scenarios and does not address automation authoring in XR environments.

X. Chen et al. (2025) identified the needs of smart homes and their contexts, and categorised them into three abstraction levels: (1) *End Users’ Intentions*, in which users describe expectations about their environment without referring to specific devices; (2) the *Intention Realisation Plan*, which includes descriptions related to device types without brand-level detail; and (3) *Device Scheduling Logic*, which requires the specification of concrete device-level instructions. Liu et al. (2023) explored the integration of natural language and multimodal interaction for IoT configuration, identifying challenges such as ambiguity, redundancy, and increasing rule complexity. Clark, Newman, and Dutta (2017) investigated how different smart home abstractions influence users’ mental models by administering questionnaires that described smart homes in different ways and analysing the resulting differences in user responses. These approaches, while effective, require users to map their intentions onto predefined abstractions (e.g. triggers, conditions, actions), which has been demonstrated to introduce cognitive overhead and formulation difficulties for non-expert users. This limitation motivates the exploration of alternative interaction paradigms, such as conversational authoring, which allow users to express their intent more freely. The increasing adoption of Large Language Models (LLMs) has also influenced research on rule-based authoring tools for EUD. Barricelli et al. (2025) examined the use of LLMs to support the creation of energy-aware smart home automations. They proposed an architecture composed of a rule checker, which interprets user input and detects conflicts with existing

automations, an energy checker, which evaluates the energy efficiency of automations, and a chatbot that mediates the interaction between these components and the user, enabling iterative refinement of user intentions within the Home Assistant platform. Similarly, GreenIFTTT (Giudici et al. 2024) employs LLMs to assist users in creating home automation routines that prioritise energy consumption optimisation. Our work shares with these approaches the use of rule-based automation and the smart home as an application domain, while distinctively targeting both AR and VR interaction. Also, although these studies address key challenges related to understanding users' natural expressions of automation goals, the systematic analysis of dialogue characteristics in interactions between users and intelligent agents remains largely unexamined in the literature. In particular, the dialogues between users and the system, in terms of e.g. their structure, recurring patterns, error types, and recovery strategies, so far have not been sufficiently analysed systematically. Our work addresses this gap by investigating how end-users define automations in natural language within AR/VR environments through conversational interaction, and by providing, to the best of our knowledge, the first systematic characterisation of the dialogues underpinning this process.

2.2. EUD authoring tools for XR

Research on EUD for XR mainly focuses on reducing development barriers to enable people without technical skills to customise or create XR environments. Torres et al. (2020) proposed an editor that supports the addition of interactive elements but is limited to 360° videos. Blečić et al. (2021) and Fanni et al. (2019) developed a desktop authoring tool for point-and-click games targeting unskilled users, which uses natural language to specify ECA rules; however, the environments are restricted to pre-captured media.

Immersive authoring tools (Adão et al. 2018; Farias et al. 2022; G. A. Lee and Kim 2009; G. A. Lee, Kim, and Billinghurst 2005; Steed and Slater 1996; Zhang and Oney 2020) have been proposed to overcome the 'build-test-fix' cycle typical of desktop-based tools, which can hinder end users during application creation. These solutions, however, are generally designed for developers. RUIS (Takala 2014) allows hobbyists to create VR experiences using building blocks, but still requires some programming knowledge. VR GREP (Zarraonandia et al. 2016), although at a preliminary stage, targets end users in the creation of VR environments, yet supports limited interaction capabilities and requires technical skills. Several EUD systems for

IoT home automation have been proposed in both industry and research, including IFTTT (Ur et al. 2014), EFESTO (Desolda, Ardito, and Matera 2017), TAREME (Ghiani et al. 2017), and ImAtHome (Fogli, Peroni, and Stefini 2017). These systems typically rely on static visual paradigms, which may complicate users' interaction with rule definitions due to limited contextual information.

Augmented reality (AR) has the potential to increase user engagement by enabling direct interaction with physical objects and supporting the dynamic discovery and modification of automations. Smartphone-based AR solutions are particularly attractive, as they leverage devices that users already own. Previous work (Knierim et al. 2019) shows that users appreciate on-demand information and enhanced perception enabled by AR in home environments.

Reality Editor (Heun, Hobin, and Maes 2013) represents one of the earliest AR approaches aimed at connecting IoT object behaviours by allowing users to map graphical elements onto physical interfaces. By linking object tags, users can define multi-object functionalities. HoloFlows (Seiger, Gohlke, and Aßmann 2019) adopts a no-code AR approach for configuring IoT workflows by connecting devices through virtual links, although it requires dedicated hardware. MagicHand (Sun et al. 2019) enables users to control IoT devices through hand gestures using HoloLens, but mainly focuses on single-device interaction. HoloHome (Mahroo, Greci, and Sacco 2019) proposes a similar AR framework, while still targeting basic use cases and offering limited support for multi-device programming. BrickAyeR (Stefanidi et al. 2019), instead, adopts a 3D building-block metaphor that allows unskilled users to define IoT behaviour using virtual blocks as triggers and actions; it also exploits a conversational agent, but it is only used as a helper and therefore does not represent their primary interaction channel; also, user evaluation remains limited. This approach was later extended in MagiPlay (Stefanidi et al. 2020), an AR-based game that enables children to programme their surroundings.

Despite all these advances, existing XR authoring tools largely rely on visual or structured interaction paradigms ranging from block-based metaphors (Stefanidi et al. 2019, 2020) to gesture-based and predominantly single-device control (Sun et al. 2019). Moreover, each tool is often confined to a single XR modality, either AR or VR, and in several cases to dedicated hardware (Seiger, Gohlke, and Aßmann 2019; Sun et al. 2019). Thus, it remains under-explored how a conversational paradigm based on natural language would reshape user behaviour in XR authoring scenarios. This work aims to address this gap by focussing on

understanding the intrinsic properties of conversational authoring in XR, realised through an agent that spans both AR on commodity smartphones and VR on immersive headsets, also supporting the orchestration of multiple objects.

2.3. LLM-based authoring of VR and AR environments

Recent research demonstrates a growing shift toward using Large Language Models (LLMs) not only for generating digital assets for immersive environments, such as images, videos, text, 3D models or textures, but also for authoring interactive behaviours in VR, AR, and broader XR systems. Across this body of work, LLMs act as conversational programming agents, co-creative partners, or runtime behaviour generators that translate natural language and embodied input into interactive scene logic.

One line of work explores LLMs as engines for interaction generation at runtime. GROMIT (Jennings et al. 2024) allows dynamic authoring of game behaviours during gameplay, by integrating an LLM with Unity to generate and compile behaviour code in response to player actions, thereby enabling emergent interactions and alternative solutions to in-game challenges. Their work demonstrates how LLMs can author interaction logic on demand, conditioned on a semantic representation of the environment. Their validation study highlights both the expressive potential of LLM-authored interactions and the need for guardrails to maintain design intent and system stability.

Other approaches position LLMs as conversational authoring tools, which allow users to create and modify interactive AR/VR experiences through natural language. ImagineAR (J. Lee et al. 2025) enables non-expert users to author outdoor AR scenes using speech-driven interaction, where LLMs interpret user descriptions and operate over scene graphs derived from real-world environments. Similarly, Aghel Manesh et al. (2024) investigate how users naturally prompt conversational agents while immersed in VR environments. Their findings show that users expect LLM-based agents to understand embodied references, maintain conversational and environmental context, and apply commonsense reasoning. Such capabilities are essential for authoring meaningful interactions in situ. In Dreamcrafter, Vachha et al. (2025) proposed an immersive editing system for VR scenes allowing users to create, position, and modify objects through natural language and direct spatial manipulation, leveraging LLMs to edit the style and appearance of objects.

Other work examines human-AI co-creation of interactions in immersive environments, emphasising controllability and agency. VRCopilot (Zhang et al. 2024) integrates generative models into VR interior design workflows, allowing users to specify layouts and object placement via speech and gestures. Although interaction logic is limited compared to full gameplay systems, the user study reveals that highly automated LLM-driven generation can reduce user agency, while scaffolded approaches (such as wireframes) help users reason about functionality and interaction before committing to detailed designs. These findings suggest that LLM-authored interactions in VR should balance automation with mechanisms for incremental refinement, supporting the inspection of the resulting artefacts. Finally, several frameworks focus on robustly translating natural language intents into interactive XR code. LLMR (De La Torre et al. 2024) decomposes natural language requests into planning, scene analysis, code generation, and inspection stages, enabling real-time creation and modification of interactive 3D scenes in Unity. J. Chen et al. (2025) propose LLMER, which avoids direct script generation by having LLMs output structured JSON representations of interactions and scene changes, reducing errors and latency in XR applications. These systems demonstrate that modular LLM architectures and structured intermediate representations are effective strategies for authoring interactions in complex AR/VR environments.

So, while previous research in this field has demonstrated the effectiveness of LLMs for authoring interactions and behaviours in AR and VR, a key distinction between these approaches and our work lies in *how* LLM outputs are represented and exposed to end-users. Most existing systems translate natural language requests, often in one shot, into executable artefacts such as scenes, assets, behaviour, which can be either opaque or otherwise difficult for non-expert users to inspect and reason about. Our approach instead relies on a single conversational paradigm operating across both AR and VR to author persistent automations expressed as event-condition-action rules and articulated in natural language, thereby increasing the transparency of the generation process. Users can review the generated rules, check whether they align with their intent, and engage in follow-up dialogue to request refinements when the outcome is unsatisfactory. Thus, we aim at creating a collaborative loop in which users can understand, modify, and adapt automations, lowering the cognitive and technical barriers traditionally associated with code-based authoring in AR and VR.

2.4. User recovery strategies from chatbot errors

Conversational agents can fail in various ways during the interaction. These errors may occur when the system misinterprets the user's intent, produces a contextually inappropriate response, or generates inaccurate information, including factual errors and hallucinations (Gnewuch and Reinkemeier 2025; Izadi and Forouzanfar 2024). In response to such breakdowns, users adopt a variety of strategies to recover from chatbot errors. Asisof (2022) showed that users prompted to rephrase mostly alter the form of their input, such as syntax, wording, length, or synonyms, rather than its informational content. Dippold (2023) found that, when an appointment-booking chatbot misinterprets users' intents, users rely not only on rephrasing, but also on strategies such as changing choices, restating, or asking question. Similar recovery strategies are also observed in interactions with LLM-based chatbots. Kim et al. (2024) show that users respond to ChatGPT errors or problematic outputs in different ways, including restating or rephrasing the prompt, pointing out or correcting errors, adapting the task or taking no further action.

These findings suggest that users try different solutions, but also highlight the need for systems that explicitly support recovery rather than leaving users to infer how to fix the interaction. Ashktorab et al. (2019) examine how intent-based chatbots can support recovery from conversational breakdowns by comparing user preferences for eight system-initiated repair strategies, including asking users to repeat themselves, requesting confirmation, deferring to a human agent in hybrid systems, and exposing aspects of the system's interpretation, such as problematic keywords. Since our analysis concerns users' recovery strategies, it can be read as the user-side counterpart to system-side repair taxonomies such as those in Ashktorab et al. (2019): their categories describe what the chatbot does in response to a breakdown, whereas ours describes what the user does once an error occurs.

Følstad and Taylor (2019) use intent recognition output to trigger a system-initiated repair strategy in a customer service intent-based chatbot, where the chatbot expresses uncertainty and suggests alternatives when a misinterpretation of the user's request is likely. The idea of supporting users through alternatives is further developed by Gnewuch and Reinkemeier (2025) who propose a collaborative repair strategy for AI-based customer service chatbots in which detected breakdowns are classified by type and addressed through repair messages combining apology, explanation, and guidance, helping users understand what went wrong and

how to reformulate their request. Braggaar et al. (2023) extend Ashktorab et al.'s work by evaluating the repeat, options, and defer strategies in the context of public-transport customer service chatbots, showing that users prefer options and defer to human agent strategies over being asked to repeat or rephrase their request. Another strategy for supporting users in recovering from conversational breakdowns is to provide explicit affordances for user-initiated repair on the agent's previous response. Moore, An, and Marrese (2024) implement seven user-initiated repair features common in natural conversation, including definition requests, indexical definition requests, understanding checks, example requests, paraphrase requests, repeat requests, and partial repeat requests. Tested with real users in three Watson Assistant scenarios, these features helped users repair understanding-related breakdowns in agent responses.

These works focus mainly on intent-based customer-service chatbots, where repair strategies are typically tied to predefined intents and help users clarify what they meant. LLM-based conversational agents can mitigate some breakdowns of intent-based/AI-based systems by better inferring user intent and handling imperfect input (Timpe-Laughlin et al. 2025). However, because LLMs can produce outputs that contain inaccuracies or hallucinations, recovery also extends to supporting users to inspect, verify, and review what the system has produced.

RELIC (Cheng et al. 2024) supports users in verifying and correcting wrong responses through the self-consistency of multiple samples generated by the same LLM. Starting from a user request, the system generates multiple responses, decomposes the main response into atomic claims, and compares each claim against information extracted from the other samples. The interface highlights whether each claim is supported, contradicted, or neutral, allowing users to inspect evidence and alternative formulations and edit the generated text. Similarly, HILL (Leiser et al. 2024) aims to help users detect potential hallucinations in LLM-generated responses by prompting the model itself to assess its generated answer (i.e. identify factual errors, retrieve source URLs, and flag problematic passages) and presenting the results through highlighted text, confidence scores, validated source URLs, and explanatory details. Sharma et al. (2026), instead, aim to reduce the likelihood that LLM-based conversational agents generate unsatisfactory or erroneous responses by using user feedback to keep them aligned with users' intents during multi-turn interactions. They implemented and tested FeedbackGPT, a conversational interface that scaffolds more targeted feedback by allowing users to annotate

specific passages, refine vague comments, and inspect how the model interpreted their feedback.

Our work builds on this literature but addresses a different form of recovery, since in conversational automation authoring the object of recovery is not only a misunderstood utterance or an unsatisfactory textual response, but a structured rule that must be complete, executable, and aligned with the user's intended behaviour. In this regard, RuleBot++ (Gallo, Paternò, and Malizia 2024) is the closest related system, as it uses GPT-4 to support the conversational creation and modification of trigger-action automations using natural language, and through rule refinement, summarisation, and confirmation before saving through a textual web interface, helping them revise the generated rule before it is stored. However, in their study, Gallo, Paternò, and Malizia (2024) analyse the errors that emerge during the interaction, while our analysis focuses on how users recover, refine, and validate automation definitions across the authoring process.

To sum up, in our study, we consider this problem in the context of XR automation authoring, where users create and revise automations with the agent's support. We combine LLM's natural language capabilities, multi-turn refinement, summarisation, and confirmation before saving with voice interaction, pointing, and gaze references, and analyse the recovery strategies users employ when chatbot errors emerge during automation authoring.

3. Wizard of Oz study

To investigate the typical patterns that end-users follow when describing their automation intent, we conducted a formative study. The goal was to identify the dialogue structure and its patterns to inform the engineering of a chatbot supporting the authoring of automations.

3.1. Method

We employed the Wizard of Oz (WoZ) method, using props to simulate both physical devices and virtual content. Through this approach, we reproduced two settings that differ in terms of interaction support (immersive VR and mobile-based AR) and application domain (a museum and a smart home). During the interaction, one of the researchers acted as the LLM-based agent (i.e. the bot), interpreting and responding to participants' questions and commands while proposing automation rules based on the dialogue. Our WoZ study, in which participants were aware that the wizard was a human, builds on prior work (Bradley et al. 2010)

reporting no significant differences in dialogue content between participants who believed they were interacting with a system and those who knew a human was operating it. This methodological stance is grounded in and supported by other studies on conversational interaction (Adler, Iacobelli, and Gutstein 2016; S. Lee and Choi 2017; Mitchell and Mamykina 2021).

We planned each test session to last approximately one hour and structured it into four phases: an interview on demographics and background, an introduction to the test, the task execution phase, and a feedback interview. In the initial phase, participants provided demographic information and details on their programming experience, familiarity with AI assistants, and prior exposure to VR/AR creation tools. We informed them about the video recording protocol for the session, and they provided their written informed consent for the study.

Six tasks followed, providing a set of automation goals for participants to fulfil by defining rules. The structure of the rules required for solving the tasks was the same across both settings in terms of complexity:

- **Familiarisation:** T1 was dedicated to familiarising with the environment;
- **Simple:** T2 and T3 required to create a simple trigger-action automation (a single event and a single action);
- **Complex:** T4 and T5 required to create a complex rule (including conditions);
- **Modify:** T6 required modifying an existing automation.

The session concluded with a short interview to gather participant reflections on the experience.

We recruited participants with no or only basic programming knowledge through convenience sampling, by sending invitations on mailing lists and social media. All of them were volunteers and did not receive any form of compensation for their participation in the study.

3.2. Analysis

We analysed the formative session using a mixed-methods approach, combining descriptive statistics on the produced automations with a qualitative analysis of the dialogues.

Collected Rules. We first analysed the set of collected rules describing the proposed automations, counting the frequencies of specific structural patterns in their definition.

Dialogue Phases After that, we conducted a *deductive thematic analysis* on the dialogue turns we recorded. This analysis allows us to discuss two characteristics of dialogues with a conversational agent for creating automations. The first is the structure of the dialogue, including the *phases* it passes through to identify the automation goal, explore different options for achieving it, iterate on the current idea, and reach a final definition. The second is the *sequencing* among these phases according to the three task types included in the user study: creating a simple automation, creating a complex automation, and modifying an existing automation.

The identification of the phases followed the six steps proposed in Braun and Clarke's framework (Braun and Clarke 2006) (data familiarisation, code generation, theme identification, theme review, theme definition and naming, report production). We selected a deductive approach to align our phase identification process to the framework proposed by Mugunthan and Gibbons (2024) in the domain of AI generation. Our hypothesis was that many of the identified phases are also applicable to rule-creation dialogues. We assigned the analysis to two HCI experts, who began by familiarising themselves with the transcript by repeatedly reading it and annotating it. Then, they independently coded each dialogue turn, consisting of a participant's question/request and the researcher's answer as the conversational agent. The two HCI experts who carried out the process had substantial agreement (Cohen's $k = .79$). They resolved discrepancies and differences in the coding through discussion until they reached consensus.

Transitions among phases. To analyse dialogue sequencing, we counted the transitions among the different dialogue phases using the labelling provided by the two HCI experts.

Dialogue Patterns. Finally, we analysed the dialogue sequencing data to identify patterns, following an iterative qualitative annotation procedure. The two experts independently marked recurring conversational structures, identifying potential candidates for the pattern taxonomy. After the independent analysis phase (Cohen's $k = .59$), the experts jointly reviewed and merged their annotations, resolving disagreements through discussion until consensus was reached.

3.3. Virtual reality museum

3.3.1. Setting

To simulate a virtual museum environment, we arranged a physical room to resemble an exhibition space. We placed small statues and images of paintings on freestanding panels to represent artworks (Figure 1,

upper part). The environment was designed to simulate automation, featuring interactive elements such as buttons, lamps, and pedestals, each marked with unique symbols for clarity. Tooltips were simulated by tagging each object, allowing participants to access information that the moderator updated during the session. The setup combined analogue and digital methods, including printed textual information, audio responses from speakers, and video displays on a tablet. This created an engaging experience similar to a virtual museum, with the researcher acting as an agent to interpret and respond to participant interactions in real time.

During the test introduction phase, participants watched a selection of videos showcasing real VR museum applications to familiarise themselves with the concept of interactive virtual exhibitions. They were then introduced to the test scenario, where they assumed the role of a museum director tasked with designing a virtual exhibition to showcase two newly acquired artefacts: a 19th-century painting and a prehistoric statuette. The simulated chatbot provided information on object states and existing behaviours while allowing participants to define new interactions. The task set followed the structure described in Section 3. We report a precise task description in Appendix A.1.1.

3.3.2. Participants

The group consisted of 14 participants (5 females and 9 males), all Italian, with an average age of 24 years ($SD = 4.09$), ranging from 19 to 33. Ten participants reported having no programming experience, while the remaining four had only basic familiarity with coding. None of the participants had prior experience with VR development tools. Regarding familiarity with conversational agents, 8 participants reported some experience using voice assistants like Siri or Google Home, while 6 frequently used text-based chatbots such as ChatGPT or Gemini. In terms of VR experience, 4 participants indicated low familiarity with VR applications, while the rest had no experience at all.

3.4. Augmented reality smart home

3.4.1. Setting

We recreated a smart home scenario by equipping an office space with several devices. Most of them corresponded to real objects, while a limited number were represented through paper-based artefacts (Figure 1, lower part). For each device, we attached a card displaying a connected-home icon, indicating both its involvement in automations and the availability of additional information. We also introduced a dedicated 'multi-purpose' panel within the office to represent simulated



Figure 1. Wizard of Oz study setup. The upper part shows the Virtual Reality museum environment, including the physical props used to simulate interaction and a participant performing the task with the researchers. The lower part presents the Augmented Reality smart home simulation, highlighting the physical devices involved and a participant during the test.

services, such as weather, location, and time, which were not tied to any specific device.

Again, the task set followed the structure described in Section 3. We report a precise task description in Appendix A.1.2. To avoid influencing how users formulated rules, we presented each task as a *scenario* that users needed to address by defining one or more automations. During the study, one researcher simulated AR functionalities by presenting device information on paper-based panels when participants framed devices with a smartphone camera, while another researcher acted as the conversational agent. Participants interacted multimodally, either by combining smartphone framing with speech or by using speech-only interaction.

3.4.2. Participants

The study involved fifteen participants (11 women), all Italian, aged between 40 and 62 years ($\bar{x} = 52$, $\sigma = 7.3$). Seven participants (out of 15)

reported no prior programming experience, while the remaining eight reported only basic knowledge. Four participants had never used voice assistants, whereas eleven reported experience with Siri, Google Home, Alexa, or Google Assistant. Five participants reported no prior experience with AR, while nine had encountered AR in contexts such as museums or games. Only one participant had previously used an open-source AR package. None of the participants had prior experience with existing automation platforms.

3.5. Results

3.5.1. Collected rules

In the simulated VR condition, participants created 112 rules, 100 of which satisfied the structural validity requirements, including 98 newly defined automations and 14 modifications. Among these, 43 automations were created in Tasks T4 and T5 alone, with 21 simple rules and 22 complex rules, compared with the 28

complex rules originally expected, indicating that several participants decomposed complex automation goals into multiple simpler rules. Twelve of the fourteen participants completed all tasks; P3 did not define rules for T4 and T5, while P5 did not complete T3. Instead, in the simulated AR condition, all participants defined at least one automation for every task, and in several cases, they produced more than one automation within the same task. In particular, users in Tasks T4 and T5 sometimes chose to create two separate rules rather than a single complex one. In T4, three participants even defined three distinct rules. Consequently, across T4 and T5 the 15 users produced a total of 47 rules (25 simple and 22 complex), rather than the 30 complex rules originally expected.

Further information on the collected rule structure is available in Appendices A.2.1 (for VR) and A.2.2 (for AR).

3.5.2. Dialogue phases

The coding process for the dialogue reveals similarities between conversational rule authoring and the interaction stages described by Mugunthan and Gibbons (2024) for AI image generation tasks, particularly in the progressive articulation and refinement of user intent. In general, users begin the interaction with vague objectives, which the assistant helps to translate into concrete, actionable events and actions. Users then explore the possible interactions and properties of the available objects to achieve their goal, iteratively adjusting the rule and specifying any condition constraints. Therefore, our analysis elicited four phases already included in Mugunthan and Gibbons (2024) (*Define*, *Explore*, *Refine*, *Export*) and required an additional one (*Confirm*) to perform a final check on the proposed automation. This additional dialogue step was necessary because the generated automation could not be visually inspected and had to be described verbally for evaluation by the user. In contrast, image generation systems allow users to directly observe the generated output throughout the interaction.

In summary, we identified the following five phases in a dialogue for defining an automation:

- (1) **Define** (D). In this initial stage, users articulate their automation goals at a high level, often using vague or abstract expressions. Rather than specifying exact triggers or actions, they convey general intentions, such as in the following request we recorded in the AR smart home scenario: *'I'd like the environment to be comfortable when I arrive'*. At this point, the conversational agent helps users translate these broad, sometimes ambiguous intents into more structured automation objectives. The experiment also elicited the need for the agent to ask clarifying questions or provide examples to bridge the gap between a user's conceptual goals and actionable automations. This emerged as a requirement for implementing an automatic agent support. Continuing the example above, after the user expresses a general desire to improve comfort, the conversational agent can respond by presenting an example of a possible way, available within the smart environment, to support that goal, by suggesting, e.g. *'You can create an automation to adjust the lighting for a comfortable atmosphere. Would you like to proceed?'*
- (2) **Explore** (E). Once a general goal is established, users enter an exploration phase, where they investigate how their intent can be realised within the system's affordances. This stage typically involves learning about the available sensors, devices, and environmental objects that can be leveraged to achieve the desired outcome. To do that, users engage in back-and-forth exchanges with the assistant, asking about possible options, capabilities, and configurations. For example, in AR, if a user expresses interest in improving home security, the agent might propose using, e.g. motion sensors, or smart locks, or an improved lighting at the house's entrance (e.g. *'You can set up an automation that turns on the entrance light when you are near home in the evening'*). In VR, users often explore interactively (e.g. by touching or manipulating objects in the environment) and ask the agent about their interaction possibilities. In both scenarios, this exploratory process helps users build an understanding of what can be automated and how. In addition, through this guided exploration, users can progressively build a mental model of the system's capabilities, gaining a clearer understanding of what can be automated and through which objects/devices, while reducing the need for users to have prior technical knowledge.
- (3) **Refine** (R). After a preliminary automation takes shape, users might need to refine the rule, ensuring it accurately and unambiguously captures the intended behaviour. Refinement involves specifying conditions (e.g. *'only if the light is on'*), spatial or temporal constraints (e.g. *'for 20 seconds'*), and contextual details (e.g. *'basically here [points to the floor], at a short distance from the artwork'*). During this iterative process, the agent actively prompts clarifications and suggests missing parameters to resolve ambiguities. In AR, users typically refine their rules through just spoken

dialogues, whereas in VR, refinement becomes multimodal: users accompany verbal instructions with gestures, pointing, or direct manipulation of virtual elements. This combination of modalities facilitates a more natural and embodied specification process.

- (4) **Confirm** (C). When the dialogue reaches a stage where the agent has gathered sufficient and coherent information, it synthesises the rule by summarising the collected parameters and asks for explicit user confirmation before proceeding. This confirmation step ensures shared understanding and allows users to verify or correct any misinterpretation. After reviewing the summary, users can either validate the rule to proceed or reopen the conversation to make further adjustments. This step serves as a checkpoint, consolidating the collaborative co-construction between human and agent.
- (5) **Export** (X). In the final stage, after the user confirms the definition, the agent operationalises the automation by saving it within the system's repository of active rules. This step marks the transition from dialogue to deployment, i.e. the moment when the collaboratively defined logic becomes part of the executable automation set. From here, users may review, activate, or later modify the stored rules as their needs evolve, completing the conversational authoring cycle.

3.5.3. Transitions among dialogue phases

Figure 2 illustrates the dynamics of conversational transitions across the five phases using a chord diagram, which visualises the relative proportions of transitions between phases in both the VR museum and AR home settings. The thickness of each chord corresponds to the percentage of transitions relative to the total number of phase changes.² The total number of analysed transitions in the VR setting is 678, while in AR they are 495.

Overall, the chord diagram reveals that the general trajectory follows a *Define* → *Explore* → *Refine* → *Confirm* → *Export* sequence, but with several local loops that evidence an iterative process rather than a strictly linear flow.

In both settings, the largest chord originates from *Define* and flows into *Explore* (VR: 7.2%, AR: 7.5%), confirming that once users have articulated a general goal, their next step is typically to discover how to operationalise it. From there, a substantial portion of the dialogue remains within *Explore* (VR: 20.9%, AR: 19.8%), indicating that users dedicate a significant share of their interaction to probing the system's capabilities, examining available sensors, and negotiating suitable

configurations. This internal loop highlights the need for iterative support to help the user's reasoning during exploration.

The next major transition links *Explore* to *Refine* (VR: 16.5%, AR: 16.6%), marking a shift from discovery to the detailed specification of automation rules. Within the *Refine* phase, the most prominent outgoing transition leads to *Confirm* (VR: 12.1%, AR: 15.6%), showing that once sufficient details have been articulated, users tend to consolidate their rules. Nonetheless, the chord diagram also reveals several backward transitions: from *Refine* to *Explore* (VR: 8.4%, AR: 1.4%), which typically occur when participants reconsider available options, and self-loops within *Refine* (VR: 6.6%, AR: 2.4%), suggesting incremental adjustments and iterative refinement before confirmation. Both patterns are more prominent in VR.

From *Confirm*, the dominant transition proceeds toward *Export* (VR: 15.8%, AR: 18.0%), completing the conversational authoring loop once users validate the automation summary. However, smaller chords connect *Confirm* back to earlier stages either to *Refine* (VR: 4.6%, AR: 0.4%) when inconsistencies are detected, or to *Explore* (VR: 1.6%, AR: 8.1%) when users reconsider their overall strategy, which happened more frequently in AR.

Self-transitions provide additional insight into how users distribute their effort. The highest self-transition rates are observed in *Explore* (VR: 20.9%, AR: 19.8%) and *Confirm* (VR: 10.0%, AR: 6.9%), indicating sustained engagement in these stages. The persistence within *Explore* suggests that users rely on iterative questioning to understand the system's affordances, while stability in *Confirm* reflects users' careful verification before committing a rule. In contrast, *Define* and *Export* exhibit minimal recurrence, consistent with their role as entry and closure phases. Together, these patterns suggest that both environments support an iterative design dialogue, with VR promoting deeper cycles of refinement and AR encouraging contextual re-evaluation before confirmation.

Overall, VR interactions exhibited higher transition diversity, reflecting exploratory and embodied behaviour, whereas AR dialogues were more compact and goal-driven, emphasising situational adaptation and efficiency in reaching confirmation.

Early-phase transitions were broadly similar across the simulated VR and AR tasks (e.g. *Define*→*Explore*: VR:7.2% vs AR:7.5%; *Explore* self-loops: VR:20.9% vs AR:19.8%), suggesting consistent conversational behaviour during the initial stages of interaction, irrespective of modality. The relatively high rate of self-transitions within the *Explore* phase (approximately 20% in both

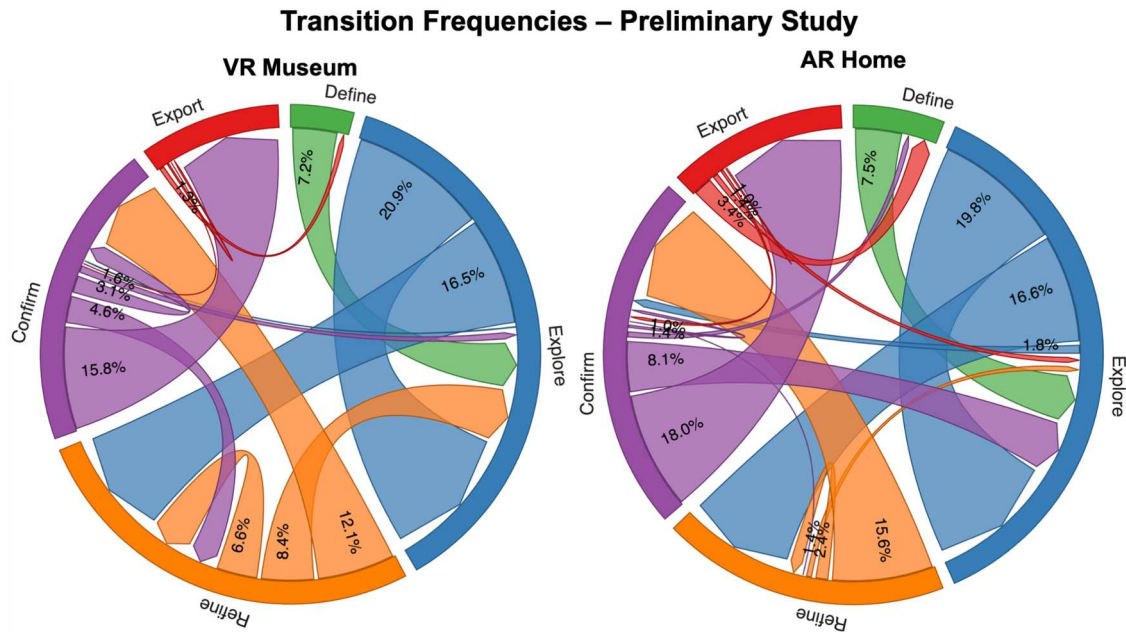


Figure 2. Chord Diagram representing transitions among the identified stages (Define, Explore, Refine, Confirm, and Export) in the collected conversations during the formative study in the VR museum (left) and AR home (right).

conditions) further indicates that participants in both contexts often required multiple conversational turns to understand system capabilities before moving to refinement. Differences emerged primarily in later phases of the interaction flow. In the simulated VR tasks, higher rates of Refine→Explore transitions (VR:8.4% vs AR:1.4%) and Refine self-loops (VR:6.6%

vs AR:2.4%) suggest a stronger tendency toward iterative refinement and re-exploration. These patterns may reflect characteristics of the simulated VR context, which emphasised object manipulation, inspection, and spatial interaction. By contrast, the simulated AR tasks showed a comparatively more direct progression toward confirmation, with fewer intermediate refinement loops

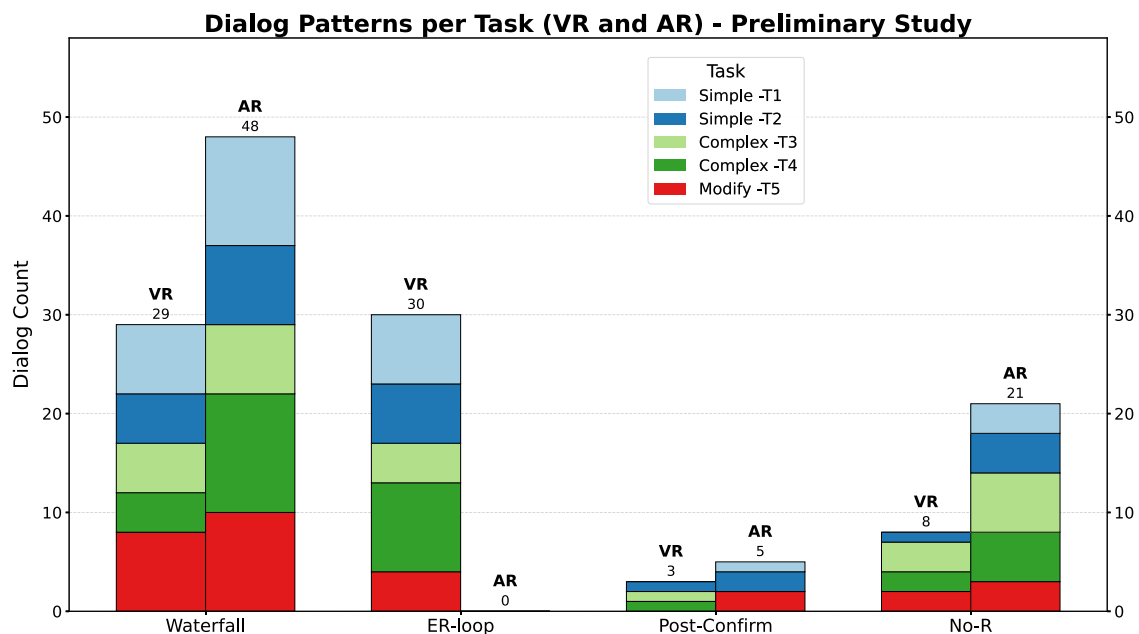


Figure 3. Distribution of automation authoring patterns in the WoZ study. The figure shows the frequency of the four identified patterns (Waterfall, ER-loop, Post-Confirm, No-R). Bars report the number of occurrences in the simulated VR (left) and AR (right) conditions, segmented by task type: simple automation authoring (bluish), complex automation authoring (greenish), and modification of an existing automation (red).

(Refine self-loops: AR:2.4% vs VR:6.6%; Refine→Explore: AR:1.4% vs VR:8.4%). At the same time, the AR-like condition exhibited a higher frequency of Confirm→Explore transitions (AR:8.1% vs VR:1.6%), whereas VR-like condition showed more Confirm→Refine transitions (VR:4.6% vs AR:0.4%). This pattern suggests that, within the AR task context, exploration was more often revisited at the confirmation stage, while the VR supported a more incremental refinement process throughout the interaction.

3.5.4. Identified dialogue patterns

Besides analysing the overall distribution of the transitions between phases, the analysis also examined the paths taken by the dialogues during the authoring of each automation, to identify recurring patterns in the sequences. We counted the occurrences of the patterns we identified for each simulated condition (VR and AR) and distinguished among dialogues for authoring a *Simple* or *Complex* automation or to *Modify* an existing one.

The analysis identified four recurring patterns and Figure 3 summarises their occurrences. It is worth noting that some patterns are not mutually exclusive, so the same dialogue may comply with more than one pattern.

Waterfall. We observed a recurring traversal of the four dialogue phases with no backtracking to earlier steps, which we term *Waterfall* by analogy with sequential, cascade-based processes. In this pattern, once participants articulate an initial definition, the conversation progresses linearly through exploration, refinement, confirmation, and export. We consider loops on the same phase or skipping one phase (e.g. jumping from *Define* to *Refine*, thus skipping *Explore*) to be compliant with the pattern. This sequencing is especially prevalent in the simulated AR home condition, where participants typically approach the authoring task with a clear understanding of the services offered by the different devices to automate. In many AR dialogues, the *define* phase already contains a nearly complete rule specification, requiring only minimal probing from the wizard before being accepted. An example is reported in Figure 4.

In contrast, when simulating the VR museum, participants needed to explore the environment to familiarise themselves with the available objects and media. In this condition, the waterfall structure is less frequently observed in T2 and T4, which are the first tasks requiring a simple and a complex automation, respectively. However, it becomes more common in the second task of the same type (T3 and T5), when participants build on their understanding from earlier interactions. The pattern also appears in many VR *modify* dialogues, which result from the last task, especially when participants applied the strategy to restate the rule from scratch rather than describe the differences from the existing one. Figures 5 and 6 are two dialogues that exemplify these situations.

ER-loop. A second recurring pattern in the collected dialogues is the *ER-loop*, characterised by at least one backward transition from the *Refine* phase to the *Explore* phase, followed by a return to the *Refine* phase. In these dialogues, participants begin by specifying a vague automation in the *Define* phase, after which the dialogue focuses on exploring and refining the options for one part of the rule (typically the trigger) before looping back to the options required to complete the other components of the automation. After that, the dialogue completes the automation specification by incrementally exploring and refining the definitions of all remaining rule parts.

We observed these cyclical dynamics exclusively in the simulated VR condition, whereas the smart home devices in the simulated AR condition provided sufficient affordances to limit the iteration of the exploration phases. VR participants engaged in exploration not only to select system capabilities but also to understand what actions are possible in the first place. By contrast, the pattern becomes less frequent in *modify* tasks (T6), where the automation to be changed already provides a clear structure and well-defined constraints, reducing the need for iterative exploration. A representative example is shown in Figure 7.

Post-Confirm. The *Post-Confirm* pattern captures situations in which participants generate new ideas or refinements *after* the system has already summarised an automation, and they have agreed to its structure.

[AR10]: I would like it to turn on automatically the purifier when the sensor detects unhealthy air.
 [Mod]: And when the air becomes healthy again? Do you want it to turn off?
 [AR10]: Exactly, but only if there's someone in the room.
 [Mod]: Great, we're working on automation. Turn the purifier on when the air isn't healthy and off when it's healthy, only if someone is present. Is that correct?
 [AR10]: Yes, perfect!
 [Mod]: Let's save the automation!

Figure 4. Waterfall pattern Example 1 (WoZ study, AR setting).

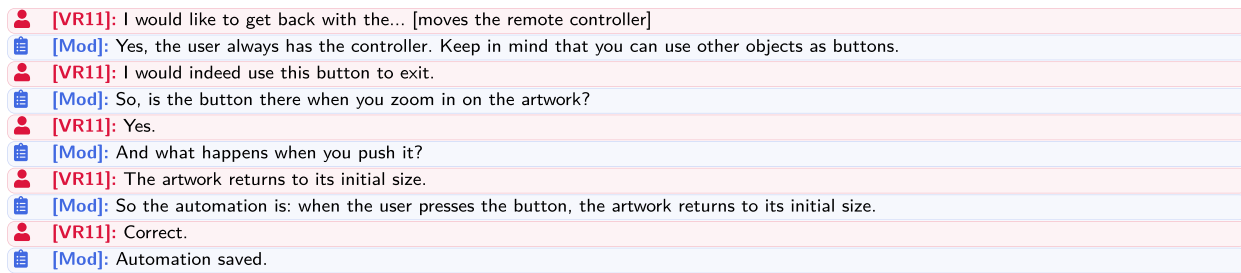


Figure 5. Waterfall Pattern Example 2 (WoZ study, VR setting).

This behaviour typically arises when hearing the summary leads users to consider edge cases or envision real-world use, prompting further ‘what if’ considerations. Figures 8 and 9 are two dialogues that exemplify these situations.

These instances demonstrate that *Post-Confirm* behaviour reflects a late-stage expansion of the rule design space, in which participants refine, extend, or add new automations only after seeing the initial one formalised by the system.

No-R. The last pattern occurs when the participant defines an automation without going through a *Refine* phase. This pattern occurred more frequently in the simulated AR smart home scenario, where the participant’s information was often precise enough to define the automation after the *Explore* phase. In the VR Museum, this occurred when the participant followed the provided suggestions without making any changes, often after some iteration of explorative turns. Figures 10 and 11 show two examples.

These data suggest that users in the simulated VR settings tend to alternate between exploratory and refinement behaviours, rather than proceeding through a more linear sequence of conversational phases: this could stem from the need to progressively discover and understand the available interaction possibilities in a less familiar context.

In both settings, the emergence of the *Post-Confirm* pattern, in which users add refinements after confirming an automation, suggests that system-generated

summaries serve as more than mere confirmations of user intent. Rather, they appear to stimulate further user reflection and prompt consideration of alternative or additional scenarios.

The different distribution of patterns across simulated AR and VR also provides empirical support for environment-dependent end-user development behaviour, showing that the same task-based automations provided through natural language dialogues can elicit distinct interaction strategies depending on the medium.

3.6. Discussion

3.6.1. Balancing simplicity and complexity in AR and VR automations

Users showed a clear preference for defining multiple simple automations rather than complex ones across both the VR and AR experiments (although this behaviour occurred less frequently in VR). Even when scenarios involved multiple triggers, conditions, and actions, participants typically decomposed them into several simpler rules instead of constructing a single, elaborate automation. This tendency confirms observations previously reported in the literature (Ko, Myers, and Aung 2004). As a connected design implication, end-user automation tools could more explicitly support rule decomposition, e.g. by offering to users the possibility to express composite behaviours as sets of simple rules rather than encouraging articulated structures.

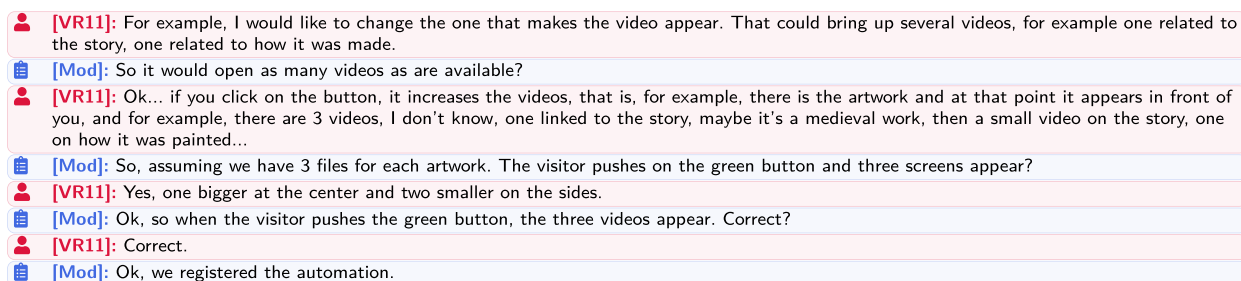


Figure 6. Waterfall pattern Example 3 (WoZ study, VR setting).

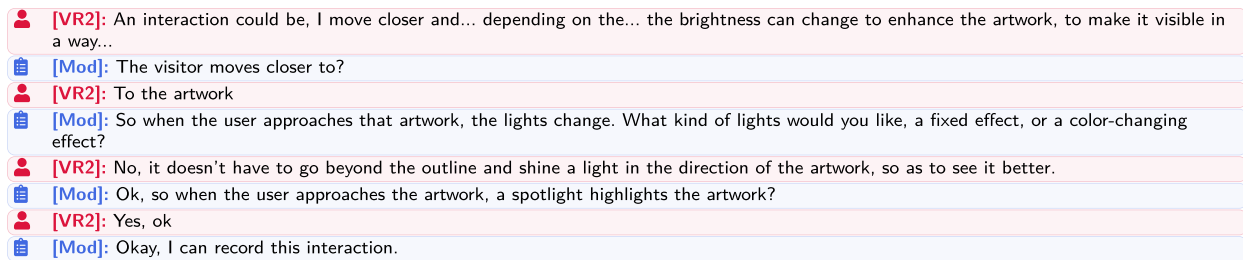


Figure 7. ER-Loop pattern Example 1 (WoZ study, VR setting).

3.6.2. Dialogue flow

The dialogue flow analysis revealed a shared high-level trajectory in both settings (*Define* → *Explore* → *Refine* → *Confirm* → *Export*), even though there are differences in iteration patterns. In both settings, we observed a prolonged engagement in the *Explore* phase, which highlights the importance of creating an agent that supports such discovery. VR museum dialogues were characterised by frequent backward transitions and self-loops within *Refine*, reflecting iterative fine-tuning and embodied exploration. In contrast, AR interactions followed a more linear progression, with fewer intermediate loops but a higher incidence of late-stage returns to *Explore* during confirmation. These findings suggest that the VR museum required a cyclical, co-constructive authoring process, whereas the AR home promoted efficiency and goal-directed behaviour, likely due to the familiarity and concreteness of smart-home contexts.

3.6.3. Familiarity with ICA

We identified clear differences between users with prior experience with intelligent conversational assistants and those without. Users in the former group expressed their intentions more clearly and succinctly, often defining

precise rules from the beginning. In contrast, users unfamiliar with such systems required multiple iterations of rephrasing and refinement before arriving at a final automation. This finding suggests that the system should support users through a guided refinement process, enabling a gradual transition from high-level, abstract intentions to more detailed and precise specifications.

3.6.4. Modifying existing rules

When asked to modify an existing automation, users frequently restated the complete updated rule instead of specifying only the element they intended to change. This behaviour may reflect limited awareness that only a part of the original rule required modification. A lack of familiarity with the logical structure underlying automations – comprising triggers, events, conditions, and actions – may have encouraged users to articulate the entire rule rather than just specific update(s).

3.6.5. Preferred modality

In the AR setting, users generally preferred voice-based interaction over camera framing, perceiving direct conversation with the agent as more intuitive. They addressed the agent either by explicitly referring to a

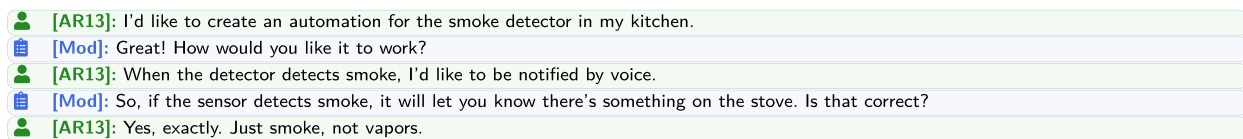


Figure 8. Post-Confirm pattern Example 1 (WoZ study, AR setting).

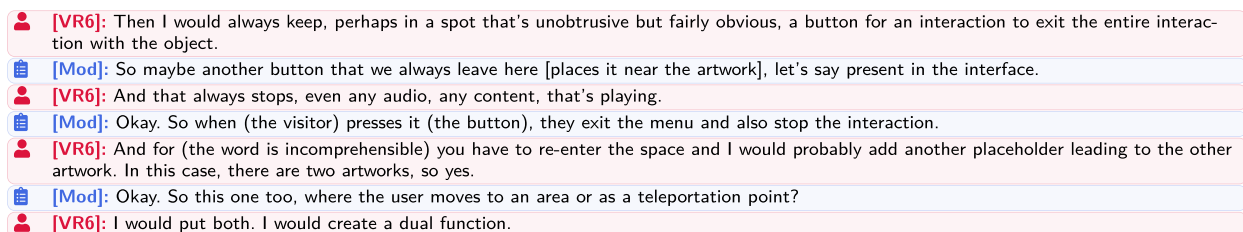


Figure 9. Post-Confirm Pattern Example 2 (WoZ study, VR setting).

device or by describing the rule's intended outcome. In contrast, VR users defined automations by interacting with environmental elements through multiple modalities. They often omitted explicit verbal references to objects, relying instead on the system's ability to infer context from the surrounding interaction. These differences indicate that the system should collect and manage contextual information in distinct ways for AR and VR environments.

4. Chatbot prototype study

In this section, we report on a follow-up study that replicates the WoZ experiment using a working chatbot prototype, Tell-XR (Carcangiu et al. 2026). We first describe its end-user features and interaction capabilities, followed by its underlying technical implementation. We then present the study conducted in VR and AR environments to evaluate the system, using tasks aligned with those from the earlier WoZ study (Section 3). Finally, we analyse the collected data to understand the characteristics of the dialogue structure.

4.1. Prototype description

4.1.1. End-user features

Tell-XR (Carcangiu et al. 2026) supports end users in the creation and modification of automations within XR environments, covering both VR and (mobile) AR scenarios. The system relies on a conversational chatbot

that interacts with users through natural language, enabling them to progress through all the identified phases of rule definition (define, explore, refine, and confirm). Tell-XR leverages GPT-4o (OpenAI 2024) together with a custom rule-processing engine to manage automation logic. It integrates speech-to-text (STT) and text-to-speech (TTS) components with additional data streams originating from VR and AR devices, thereby supporting multimodal interaction. The system implements stateful dialogue management, allowing users to update existing automations based on their inputs, and connects to XR environments via APIs to execute automations in both VR and AR contexts.

The main features and capabilities of Tell-XR are the following:

- (1) *Creation and modification of automations*: Users define and update automation rules by conversing with the chatbot. The dialogue follows a structured but adaptable flow that guides users through the rule-definition phases discussed in Section 3.6.2.
- (2) *Multimodal interaction*: Voice represents the primary interaction modality, while the system also exploits contextual cues from other modalities to enrich communication with the user.
- (3) *Context-aware interpretation*: The chatbot keeps track of the user's context, including information about the current VR/AR environment and the user's ongoing activities. This contextual awareness supports more precise interpretations of user

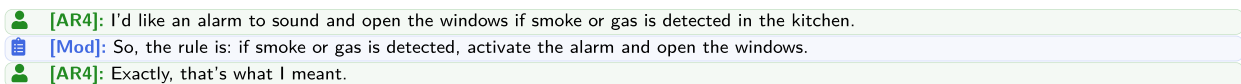


Figure 10. No-R pattern Example 1 (WoZ study, AR setting).

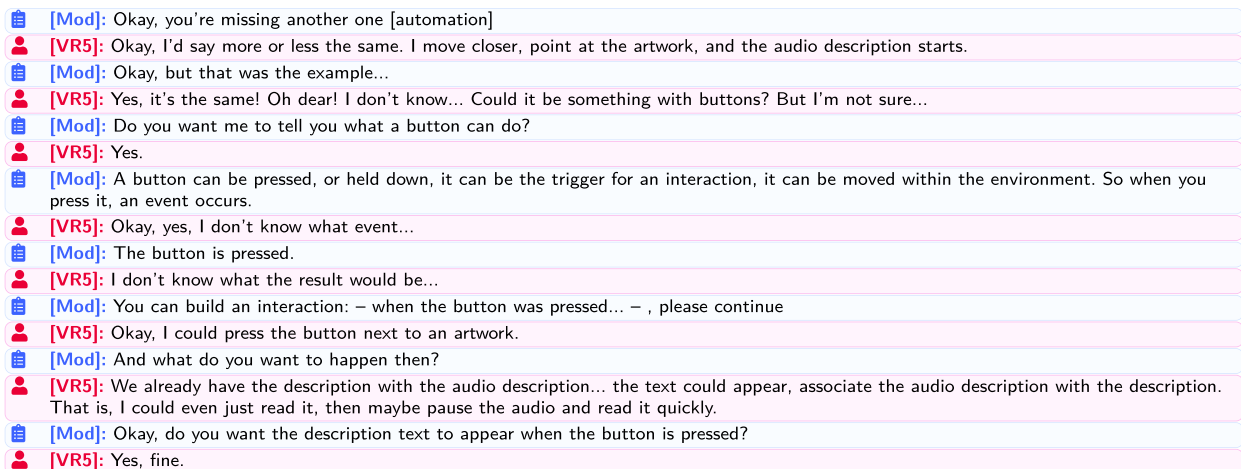


Figure 11. No-R pattern Example 2 (WoZ study, VR setting).

requests and enables the chatbot to suggest relevant objects, conditions, or actions based on previous interactions.

- (4) *Platform-specific support for VR and AR*: In VR scenarios, the chatbot enables interaction with virtual objects, whereas in AR it operates on real, physical smart devices. In both settings, the system accesses structured descriptions of available objects and devices to provide tailored suggestions during automation definition.

4.1.2. Technical implementation

Tell-XR adopts an *agentic architecture* (Figure 12) in which the conversational workflow is distributed across a set of specialised agents, each responsible for a distinct aspect of rule authoring. Tell-XR implements six agents. Five correspond to the identified dialogue phases in the WoZ study:

- The *Define* agent interprets vague or underspecified descriptions and formulates an initial rule proposal, grounded through systematic queries to contextual tools.
- The *Explore* agent helps users understand which objects are available, their capabilities, and possible property values, using gaze, pointing, and explicit references.
- The *Refine* agent requests missing details, translates vague statements into explicit parameters, checks validity, and suggests constraints.

- The *Confirm* agent summarises the current automation in natural language and requests explicit approval before saving.
- The *Export* agent converts the confirmed rule into the JSON representation required by the automation engine, updating or creating rule identifiers as needed.

The first four agents operate only at the conversational level and do not persist changes to the engine that manages the automation in the XR environment. Their role is to help the user progressively articulate a complete and meaningful rule. The export agent generates and saves the final automation structure, ensuring that intermediate or incomplete ideas do not affect the system state.

To manage the conversational flow, Tell-XR employs a *Routing Agent* that determines which specialised agent should respond next, based on the user's most recent utterance, the current state of the partially defined automation, and the interaction context from the XR environment. Each destination agent is associated with a transition prompt that encodes when the system should shift to one of the dialogue phases. High-level or vague intents redirect the dialogue to *Define*. Questions about objects, deictic expressions, or gestures trigger the *Explore* agent, while parameter adjustments or missing details activate the *Refine* agent. When the automation appears complete, routing moves to *Confirm*, and only after explicit user approval, it proceeds to *Export*.

It is worth noting that in the *Confirm* phase, the defined rule is presented to the user as a sentence expressed in natural language, whereas the rules generated in the *Export* phase are true automations that can be executed by the *Home Assistant*³ engine (in other words: they are not intermediate code). Furthermore, the natural language sentence provided in the *Confirm* phase is semantically equivalent to the automation delivered as output of the *Export* phase.

All Tell-XR agents share a common behavioural foundation: each can draw on contextual tools that provide real-time information about the environment—including what the user is looking at, which objects are nearby, and what capabilities these objects expose. Agents also have access to the full conversation history since the last confirmation step, enabling them to interpret user intent with continuity rather than relying solely on the latest utterance.

All agents adopt a consistent communication style: short, simple, and non-technical language to support non-expert users through incremental rule authoring.

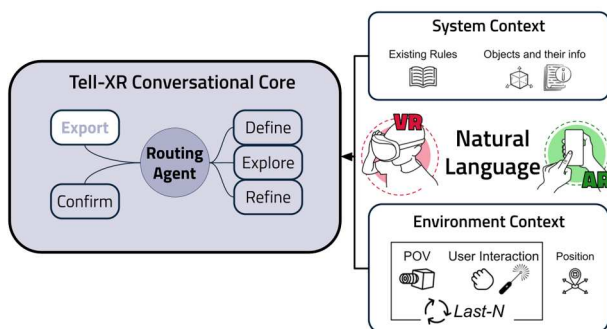


Figure 12. Conceptual overview of Tell-XR. Users interact through natural language and embodied actions in VR and AR (Mobile) environments. The system dynamically routes (Routing agent) the conversation across distinct dialogue phases (Define, Explore, Refine, Confirm), supporting iterative and context-aware automation authoring. Interaction is grounded in contextual information that is both external (from the environment, e.g. entities gazed at or manipulated by the user and their spatial positions) and internal (within the system, e.g., existing rules, available entities, and their capabilities). The Export phase persists the resulting ECA automation to the execution engine (not shown for readability).

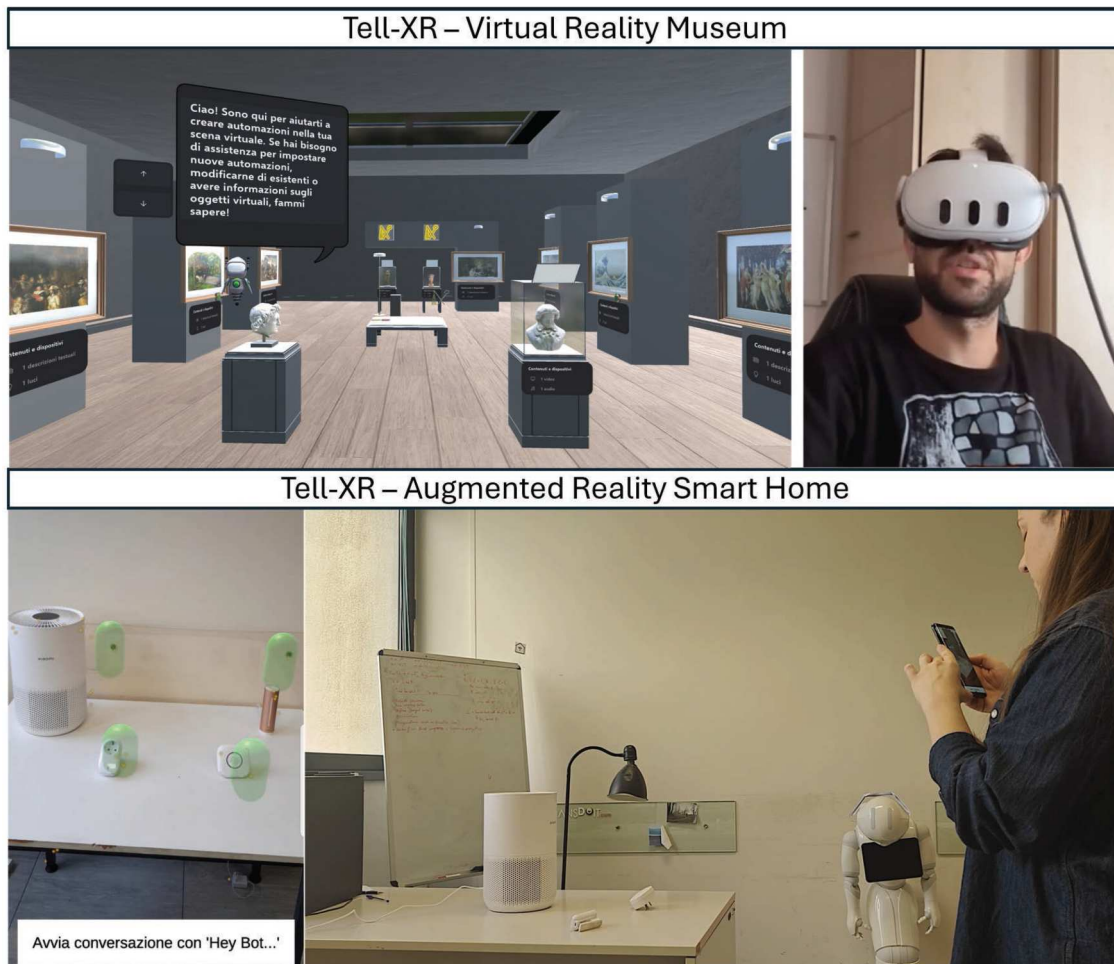


Figure 13. Chatbot prototype study setup. The upper part shows the Virtual Reality museum environment, while the lower part shows the Augmented Reality smart home.

Responses are designed to be facilitative, typically ending with an open question that encourages conversational progress while still allowing users to redirect the focus freely. All agents were prompted to remain responsive to user intent, grounded in the available scene context, and robust to the partial, ambiguous, or rapidly shifting instructions that characterised the automation authoring dialogues recorded in our WoZ study.

4.2. Method

The goal of the study was to analyse the structure and find patterns in the resulting dialogues, identifying the types of errors occurring in the automation authoring and the resulting recovery strategies.

To achieve these goals, we replicated the structure of the Wizard-of-Oz experiment in Section 3 but this time Tell-XR supported the dialogue-based authoring with the end-users. We created a prototype virtual museum for the VR setting and we placed different smart devices

in our research lab to simulate a smart-home to automate through a mobile AR app.

Before starting the study, participants received an explanation of the research objectives and procedures and signed an informed consent form covering data handling and recording. Underage participants provided parental consent.

The session consisted of three phases. In the first one, we asked participants to provide demographic information (gender and age) and to provide an assessment of their familiarity (in a 1–5 scale, where 1=very low and 5= very high) with voice assistants, XR applications, automation systems and programming. After that, we provided them with documents and videos explaining how to interact in the two environments. When the familiarisation part was finished, participants started executing the tasks phase, which had the same structure and difficulty across AR and VR.

- **Simple:** T1 and T2 required participants to create a *simple automation* (1 event + 1 action).

- **Free:** T3 asked users to design an automation of their choice (*free*).
- **Complex:** T4 and T5 involved the creation of a *complex automation*, requiring conditions and one or more actions.
- **Modify:** T6 asked users to *modify* the (simple) rule created in T2 to make it complex by adding a condition.

Unlike the other tasks, which described a specific high-level situation to address (e.g. improving safety, controlling lighting, or regulating environmental conditions), T3 intentionally left the automation objective open-ended, allowing participants to freely decide both the purpose of the automation and the strategy to implement it. This task was designed to elicit more spontaneous, exploratory, and self-initiated conversational behaviours.

During the tasks, we collected all interactions with the environment and the dialogues with the chatbot through a detailed log, including 1) the automation resulting from the authoring process, 2) the task time, 3) the transcript of the participants' vocal requests to the chatbot, 4) the chatbot responses, including the agent that was responsible to generate the answer (see Figure 12), 5) the interaction events in the environment.

Recruitment occurred through mailing lists associated with the authors' institutes and their broader research network. For the VR setting, we also contacted students from a collaborating high school. Participation was voluntary and uncompensated. All participants provided informed consent for participation and recording.

4.3. Analysis

We briefly describe here how we analysed the data collected during the study. We assigned all the analyses requiring labelling the interaction logs to two HCI experts who run the process independently. After this first step, they discussed the discrepancies in their labelling until reaching a consensus. We report Cohen's kappa as a measure of the agreement after the independent annotation round.

Completion Time. We analysed the time recorded in the logs using descriptive statistics.

Task Success. We evaluated task success by assigning each rule to one of six correctness levels, from fully correct (*score* = 6) to fully incorrect or not generated at all (*score* = 1). We report the definition of the levels in line with the results discussion for readability. We established the levels before running the study and the two experts went through the logs and assigned the success level to each task (Cohen's $k = .73$).

Errors. Error analysis followed a multi-stage qualitative annotation procedure. In the initial phase, two experts independently inspected the outputs, identified the errors, and assigned open-ended descriptive tags to characterise the observed error types. In the second phase, the annotators collaboratively reviewed the emergent tags and developed a shared taxonomy of error categories. In the final phase, both experts independently labelled each error according to the agreed taxonomy (Cohen's $k = .68$), and annotation discrepancies were resolved through adjudication and consensus discussion.

Recovery Strategies. The analysis of the recovery strategies followed the same approach we used for the errors.

Transition among Phases. We counted the transitions among the different dialogue phases by using the labelling computed by the Tell-XR routing agent.

Dialogue Patterns. The dialogue analysis followed an iterative qualitative annotation procedure. First, the two experts independently reviewed the collected dialogues and annotated instances that matched the patterns identified in the previous study. In parallel, the annotators also marked recurring behaviours and conversational structures not covered by the existing taxonomy in order to identify potential new patterns. After the independent analysis phase (Cohen's $k = .65$), the experts jointly reviewed and merged their annotations, refining the taxonomy where necessary and resolving disagreements through discussion until consensus was reached.

4.4. Virtual reality museum

4.4.1. Setting

The Virtual Museum represents a fictional cultural heritage exhibition implemented in VR, where users assume the role of a museum curator (Figure 1, upper part). The environment includes multiple rooms displaying artworks enriched with multimedia elements such as informational panels, videos, and audio narrations that curators can activate to enhance visitors' experiences. Some artworks include interactive components, for instance lighting systems or buttons that trigger specific behaviours. The museum also provides tables with supplementary materials that users can freely arrange to customise the exhibition layout. Interaction with the virtual environment occurs through VR controllers, which allow users to move objects, manipulate elements, and navigate menus.

The VR study took place either in our laboratory or at a collaborating high school. Participants explored the Virtual Museum using a Meta Quest 3 or 3S headset.

During the familiarisation phase, participants watched an instructional video explaining how to interact with the virtual environment. They subsequently explored the museum to become aware of the objects' states and the automations available. Participants then received a paper-based description of six tasks with increasing complexity. Five tasks required creating new automations, while one focussed on modifying an existing one. The complete list is available in Appendix A.3.1.

4.4.2. Participants

The study involved twelve participants (6 women) aged between 16 and 43 years ($\bar{x} = 26.2$, $\sigma = 10.0$). All participants were students, humanities professionals, or enrolled in courses related to cultural heritage promotion.

Familiarity with voice assistants (e.g. Siri, Alexa) was moderate: seven participants selected level 3, and two selected level 4. Four users reported knowing Siri, five Alexa, one Cortana, five ChatGPT, and one Google Assistant.

Participants showed limited familiarity with XR applications: eight selected level 1, three level 2, and one level 3. None mentioned specific VR applications, referring instead only to headsets. Knowledge of automation systems was similarly low; only one participant selected level 4 (Google Home, Samsung SmartThings), while all others selected level 1. Programming experience was minimal, with a single participant reporting basic HTML knowledge.

4.5. Augmented reality smart home

4.5.1. Setting

We conducted the AR test in a laboratory environment configured to simulate a smart home populated with multiple smart devices and sensors (e.g. a washing machine, smart light, smart plug, and gas sensor). Participants could move freely within the space and interact with the agent at any time using a Samsung Galaxy S10 smartphone, either through voice input or by framing a device with the camera. Camera framing enabled AR-based access to contextual information about devices, including their current state, capabilities, and existing automations (Figure 13, lower part).

Participants first completed a familiarisation phase, during which they learned about the chatbot and explored the available smart devices and their functionalities. After that, they completed the tasks reported in Appendix A.3.2.

4.5.2. Participants

Thirteen users (10 women), aged between 30 and 64 years ($\bar{x} = 48.2$, $\sigma = 11.1$), participated in the AR study. Users reported no familiarity with smart or conversational agents, three reported low familiarity, and five reported moderate familiarity ($\bar{x} = 2$, $\sigma = 0.9$). Participants reported experience with Alexa (7 users), ChatGPT (2), Siri (2), and Google Assistant (1). Most participants (12) reported no familiarity with automation systems; one reported limited experience with TaDo. Programming experience was minimal: twelve participants reported none, and one reported basic HTML knowledge.

4.6. Results

4.6.1. Collected rules

In the VR setting, participants created 86 rules. We observed a preference for simple event-driven structures, with most rules following event-action formulations. More elaborate structures combining events, actions, and conditions appeared only occasionally. At the same time, a relevant portion of rules were centred primarily on actions or conditions rather than explicit triggers, indicating diverse conceptualisations of automation tasks. Overall, the results suggest that participants favoured simple causal relationships while still exploring a broader variety of rule structures than previously reported.

In the AR setting, participants created 70 rules, evenly split between simple and complex formulations, with users accepting the chatbot's suggestions without modification in 20 cases, particularly in more open-ended tasks. Structurally, action-oriented formulations were the most common, indicating that participants primarily focussed on desired outcomes rather than triggers.

We provide a detailed report on the collected rule structure in Appendix A.4.

4.6.2. Completion time

Table 1 reports the completion time for each task in both settings. Task completion patterns reflected both task complexity and increasing familiarity with the systems. In both studies, users required more effort during the initial learning phase and completed later tasks more efficiently. Tasks involving open-ended goals or complex conditions were more demanding, particularly in the VR museum, while rule modification tasks were generally easier than creating complex rules from scratch.

While we do not have reference points for VR, the completion times in AR are about 35% longer than

Table 1. Task-level performance in the VR and AR settings with Tell-XR.

	Task	Time		Rounds			Task	Time		Rounds	
		$\bar{x}$ (mm:ss)	σ (s)	$\bar{x}$	σ			$\bar{x}$ (mm:ss)	σ (s)	$\bar{x}$	σ
VR	1	5 m 21	154	6	2	AR	1	4 m 13	148	10	6
	2	3 m 44	89	4	2		2	3 m 47	156	9	4
	3	7 m 25	146	9	3		3	3 m 34	160	10	5
	4	6 m 44	216	7	4		4	2 m 21	64	7	3
	5	4 m 42	189	6	2		5	2 m 46	60	8	3
	6	4 m 40	167	5	2		6	3 m 21	104	10	3

Notes: Each condition comprised the same six tasks. For each task, the table reports the mean completion time (in minutes and seconds) with its standard deviation (in seconds), and the mean number of conversational rounds (i.e. user–chatbot turns) with its standard deviation.

those observed using visual authoring in similar mobile AR tasks (Ariano et al. 2023). Considering the time requested for interpreting the vocal input and the LLM latency, the authoring requires an acceptable time.

4.6.3. Task success

The analysis of the errors that occurred during the studies assigned the following scores (we provide examples for each level in Appendix 3):

- **Score 6:** Both the user's intent and the generated automation are fully correct with respect to the task requirements.
- **Score 5:** The chatbot correctly implemented the user's request, but the user-defined automation is only partially correct for the task.
- **Score 4:** The user correctly specified the intended automation, but the chatbot generated an automation that is only partially aligned with the user's intent.
- **Score 3:** The chatbot correctly implemented the user's request, but the user's request itself is incorrect with respect to the task requirements.
- **Score 2:** The user specified a correct automation goal, but the chatbot generated an incorrect automation.
- **Score 1:** Both user intent and chatbot response are incorrect, or no rule is created due to misunderstandings and lack of confirmation or clarification.

In the VR museum, scores indicate high overall task success. Fully correct rules (score 6) occurred 36 times out of 72 rules. Users specified correct intents 12 more times (score 4). The chatbot was able to fix partially correct user input 10 times (score 5) and wrong input 7 times (score 3), mainly during modification and complex tasks. Scores 1 and 2 were less frequent, respectively 6 and 1 cases, indicating limited severe misunderstandings.

In the AR smart home, fully correct rules (score 6) were obtained 34 times out of 78. Users expressed correct intents in 12 additional cases (score 4), where the chatbot introduced minor omissions or hallucinations. The chatbot correctly handled incomplete user instructions 10 times (score 5), and even managed to properly

execute incorrect inputs in 7 cases (score 3). Score 2 was less common, with only one case. Score 1 appeared 10 times, 8 of which produced no rule. This was caused by user or system errors. These failures involved 6 users and occurred mainly during complex or modification tasks.

User-related errors emerged as the main source of incorrect outcomes, particularly in tasks requiring complex automations or rule modification, suggesting the need for stronger guidance in cognitively demanding scenarios.

4.6.4. Errors

Figure 14 shows the occurrences we recorded for each type of error, categorised by task. In total, we had 143 errors in the VR Museum (11.92 errors per session on average, max = 29, min = 3), while we found 179 errors in the AR home setting (average 13.76, max = 26, min = 7). However, they did not necessarily cause severe failures (only in 8 cases did no rule be created from both chatbot and user errors, all occurring in the AR scenario).

During the analysis, different types of errors emerged: those independent of both the user and the chatbot (*voice recognition errors* - RE), those made by the chatbot (*chatbot errors* - CE), and those made by the user (*user errors* - UE).

The most frequent error in the VR museum was the 'voice recognition' error.

Among the chatbot errors, we distinguish between (i) *inconsistent/wrong chatbot response* - WC and (ii) *hallucinations* - HC, which are plausible but false chatbot responses. In both settings we had more WC errors than hallucinations.

Among the user errors, there are three subtypes: (i) *user digressions* - UD utterances not directly related to the task; (ii) *wrong user response* - WU the user answer led to an error in, e.g. solving the task; (iii) *inconsistent user response* - IU, where the user answer was not coherent with the chatbot's request. In the VR museum, we counted 33 user errors in total. In the AR home, the total number of user errors is 100. We noticed that

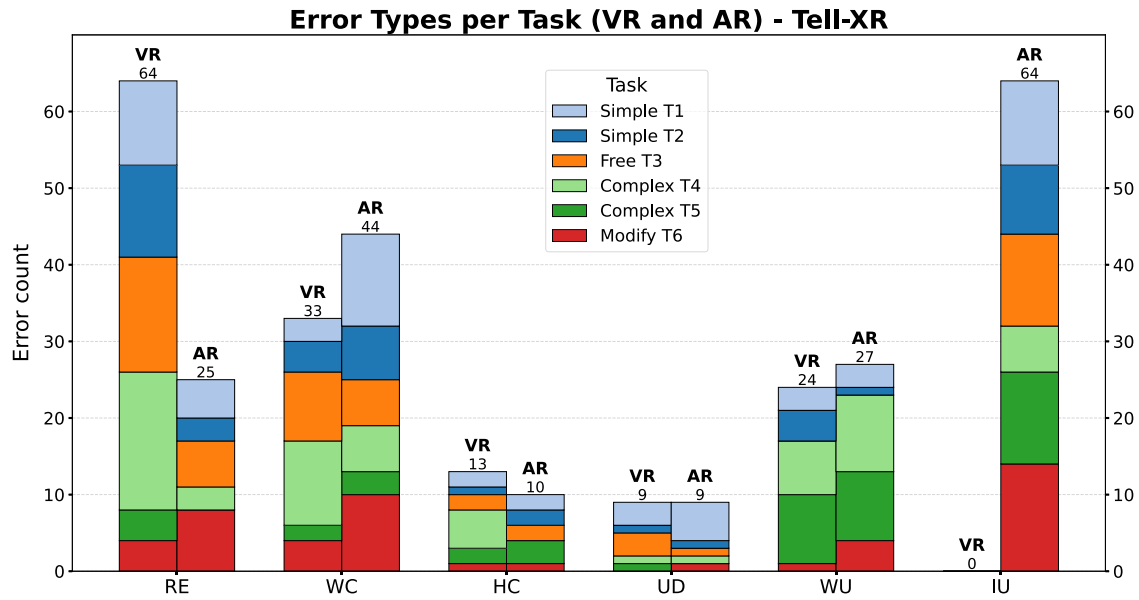


Figure 14. Visual distribution of system and user errors in automation authoring dialogues with Tell-XR. System errors include recognition errors (RE), wrong chatbot answers (WC), and hallucinated answers (HC), while user errors include user digressions (UD), wrong user answers (WU), and incoherent user answers (IU). Bars report the number of occurrences in the VR (left) and AR (right) settings, segmented by task type: simple automation authoring (bluish), complex automation authoring (greenish), and modification of an existing automation (red).

user digressions contributed to inconsistencies but never affected the final rule correctness.

4.6.4.1. VR Museum. In the VR Museum setting, we recorded a total of 22 errors for T1 (*Simple*), half of which were related to the voice recognition system (11). In general, system-related errors occurred in 16 instances. In addition to voice recognition errors (RE), which were the most frequent (11), we also observed 3 occurrences of chatbot wrong responses (WC) and 2 chatbot hallucinations (HC). User-related errors occurred 6 times, equally distributed between wrong responses (WU) and digressions (UD). In T2 (*Simple*), for the VR prototype, we counted 22 errors. The errors were mainly related to the system (77%): the voice recognition produced 12 errors, followed by chatbot wrong responses (4) and hallucinations (1). We observed 5 occurrences of user errors, mainly due to incorrect responses, with only one case of user digression.

In the first task, errors were distributed across participants, but in the second task, 3 out of 4 incorrect responses occurred in the same participant. This may indicate that the other users adapted to the system after completing the first task. Meanwhile, system-related errors remained relatively stable, with a slight decrease in HC (from 2 to 1) and a slight increase in WC (from 2 to 3).

T3 (*Free*) was the second-highest in error count. The system remained the main source of errors (26): 15 for

the recognition system, 9 for chatbot wrong responses and 2 chatbot hallucinations. We recorded 3 occurrences of user-related errors, all of which were user digressions. User wrong responses were absent, making it the only task without this error type.

T4 had the highest number of errors, totalling 42. We observed 34 system-related errors, in which the voice recognition system failed 18 times, followed by chatbot wrong responses (11), and hallucinations (5), representing the highest frequencies observed for each category across all tasks. User-related errors count 7 wrong responses and one digression. In T5, we observed 18 errors, with user-related errors surpassing system-related errors: 9 wrong user responses and one user digression. In terms of system-related errors, the voice recognition system reached its minimum frequency (4), incorrect chatbot responses and hallucinations occurred both twice.

On the one hand, in T5 we observed a decreasing trend in the number of system-related errors compared to T4 (8 vs 34). Thirteen of the 34 errors in T4 were noticed in the same participant, 8 RE, 3 HC, and 2 WC; nevertheless, T4 remains the task where the chatbot produced the highest number of incorrect responses. On the other hand, T5 appears to be slightly more challenging for users, showing a modest increase in user errors compared to T4 (10 vs. 8).

Regarding T6, only 10 errors were observed, 9 of which are system-related. Voice recognition errors

and incorrect chatbot responses each occurred 4 times, while chatbot hallucinations dropped to their lowest level, with only 1 instance recorded. User-related errors accounted for the remaining 10% of the total, consisting of a single incorrect user response, and no user digressions were observed.

Aggregating the observed values by task type, simple and modification tasks showed the fewest wrong chatbot responses (WC, 7 and 4, respectively), whereas in free and complex tasks we noticed the highest number of WC (9 and 13). This suggests that, as expected, the chatbot has more difficulty guiding users when defining automations more complex than trigger-action and when users do not have a clear idea of what to automate.

Hallucinations were primarily observed during complex tasks (7), while they were comparatively rare in simple, modification, and free tasks (respectively 3, 2, and 1). In particular, in T6 the system reached an optimal performance level, also confirmed by the reduced number of HC and RE (respectively 1 and 4). We expected hallucinations to occur at similar rates across all task types. Although *Modify* tasks involve fewer turns, which affects their hallucination count, the other task types have comparable turn counts. Overall, the results suggest that the chatbot is more prone to hallucinations when defining rules with conditions or multiple actions.

Regarding user-related errors, the complex tasks resulted in the highest number overall (18), consisting of 16 WU and 2 UD, confirming them as among the highest in cognitive demand for users. In the simple tasks, we noticed fewer errors (11), which were more evenly distributed between WU (7) and UD (4), followed by the modification task (3 UD and 0 WU) and the free task (1). The progressive reduction in digressions suggests that users gradually adopted a more focussed authoring strategy. The low error rate observed in the modification task, despite the difficulties identified in the WoZ study, indicates that participants became sufficiently familiar with the chatbot to overcome this barrier.

4.6.4.2. AR Smart Home. The errors per task in the AR smart home scenario are summarised in Figure 14. In T1, errors are distributed equally between system- and user-related failures. System-related errors occurred 19 times, with chatbot incorrect responses being the most prevalent (12), followed by voice recognition errors (5) and chatbot hallucinations (2). User-related errors totalled 19 occurrences. Within this category, user incoherent responses were the most common, accounting for 11, while user digressions resulted in 5 errors and

user wrong responses accounted for 3. Considering that this was the first task, the high incidence of user-related errors may reflect an initial learning phase as participants had to familiarise themselves with the interaction paradigm.

T2 generated 23 errors, with system-related errors occurring in 12 instances. Chatbot incorrect responses were the most frequent system failure (7), followed by voice recognition errors (3) and chatbot hallucinations (2). User-related errors were 11, with a higher occurrence of incoherent responses (9), while we found 1 instance of both user digressions (UD) and user wrong responses (WU).

T3 had a total of 27 errors, with system-related errors accounting for 14 occurrences. Chatbot wrong responses and voice recognition errors were the most frequent failures. User-related errors were 13; the 12 incoherent responses accounted for the majority of these errors, suggesting that some users had difficulty maintaining coherent interactions. User digressions (UD) remained minimal (1), while user wrong responses (WU) were absent.

T4 had 26 errors in total. System-related ones occurred 9 times, predominantly due to chatbot's wrong responses (6). User-related errors were the most frequent (17 occurrences): they included 10 wrong user responses, 6 incoherent user responses, and 1 user digression.

T5 resulted in 28 errors. System-related errors were less frequent (6 occurrences): in particular, 3 were wrong chatbot responses, and 3 were chatbot hallucinations. No voice recognition error occurred. User-related errors appeared 22 times: user incoherent responses occurred 13 times, user wrong responses 9 times, while user digressions were absent.

T6 generated a total of 38 errors, tying with T1 for the highest count across all tasks. System-related errors occurred 19 times: chatbot wrong responses occurred 10 times, voice recognition errors occurred 8 times, and chatbot hallucinations decreased to minimal levels (1). User-related errors occurred 20 times: incoherent responses occurred 14 times, user digressions remained minimal (1), and user wrong responses occurred 4 times. The higher error rates for both system and user categories suggest that T6, which required modifying an existing rule, presented one of the most challenging interaction scenarios.

When analysing augmented reality (AR) errors by task type, we notice a shift from system-related errors to user-related errors as the complexity of the rules increases. For simple tasks (T1–T2), we recorded 61 errors, divided into system-related (31) and user-related (30). In contrast, complex tasks (T4–T5) resulted in a

similar total number of errors (54), with the majority (39) being user-related failures, compared to only 15 system-related failures. The free-authoring task (T3) fell between these two extremes, resulting in 27 errors: 14 system-related and 13 user-related. The modification task (T6) had the highest overall error count (38), featuring a nearly balanced split between system and user errors (18 vs 20).

System-related errors in AR were primarily connected with incorrect chatbot responses. These types of errors were most common in simple tasks, accounting for 19 errors, followed by modification tasks (10) and free tasks (6), and were least common in complex tasks (9). Voice recognition errors concentrated in free and modification tasks (6 and 8, respectively), while we counted only 3 in complex tasks and 8 in simple tasks (T1+T2). This suggests that open-ended specifications and iterative editing may lead to longer, more variable, or less constrained utterances, thereby challenging speech input recognition. Hallucinations were relatively infrequent across AR and did not increase with complexity: simple tasks had 4; free tasks had 2; complex tasks had 3; and modification tasks had 1. This contradicts expectations that multiple conditions and actions would trigger more hallucinated content.

User-related errors clearly differentiate between simple and complex authoring tasks. In complex tasks (T4–T5), the predominant type of user error was incorrect responses, which counted 19 and significantly surpassed those in simple tasks (4), the free task (0), and modification tasks (4). This suggests that complex rule construction predominantly challenges users' ability to provide accurate details. Incoherent user responses were substantial across all task types (simple: 20; free: 12; complex: 19; modify: 15), but their relative impact was most notable in free and modification tasks, in line with the linguistic variability and decision-making demands that come with deciding what to automate or revising existing rules. Lastly, digressions were most common in earlier simple tasks (6 out of 9 total UD across all tasks), indicating a progressive reduction in this type of error as participants adapted to the interaction paradigm.

4.6.5. Dependencies among errors

An analysis was also conducted to determine the extent to which errors occurred independently or as a consequence of previous failures.

We first analysed whether user-related errors occurred independently of, or as a consequence of, previous system errors. In AR, user failures occurring without any involvement of the chatbot were 95 (out of 100 total user errors), while the remaining failures (i.e. those

triggered or influenced by preceding errors) occurred only 5 times. Thus, in AR, the vast majority of user failures were attributable to individual factors rather than to cascading effects from system malfunctions. In the VR prototype, 6 out of the 25 user wrong responses (dependent WU) were influenced by system errors. Specifically, 4 of these errors were tied to incorrect responses from the chatbot (WC), while the remaining 2 were due to chatbot hallucinations (HC). In contrast, independent wrong user errors (independent WC), which occur without any system involvement, totalled 18, accounting for 75% of the errors.

Then, we did the same analysis for system-related errors. In VR, we found that about half of the incorrect responses from the chatbot (15 out of 33) were linked to other errors. Eleven of these incorrect responses were influenced by previous chatbot errors (other WC), three by voice recognition issues (RE), and one by hallucinations (HC). This suggests that errors could propagate, leading to additional mistakes. Additionally, when examining chatbot hallucinations, we discovered that 5 out of 13 cases (dependent HC, $\sim 38\%$) were associated with voice recognition errors (RE), followed by two instances related to user wrong responses (WU). These findings suggest that the chatbot may benefit from a more robust semantic validation mechanism to prevent the propagation of errors.

In the AR home scenario, the majority of system errors (63 out of 79, $\sim 80\%$) occurred independently of preceding failures, while a smaller proportion (16 out of 79) were triggered by prior errors. Thus, a limited proportion of failures came from error propagation. The following dialogue example (associated with AR13 user while completing T1) shows how a single (independent) hallucination compromised the task outcome by generating a cascading error: the chatbot initially hallucinates about a (non-existent) 'shut off gas' action (classified as an independent system error), and this erroneous behaviour propagated through next turns until the system in the end (falsely) confirmed the successful creation of an automation that includes that non-existent action (thereby classified as a 'dependent' system error).

Overall, from the recorded data, we can see that both user and system errors had higher dependency percentages in VR than in AR. In addition, within each condition, system errors had higher dependency rates than user errors (in AR: 20% of system errors were dependent, vs. 5% of user errors; in VR: 45% vs. 25%). This suggests that system errors tend to be more prone to cascading effects than user errors, and this situation is more pronounced in fully immersive environments.

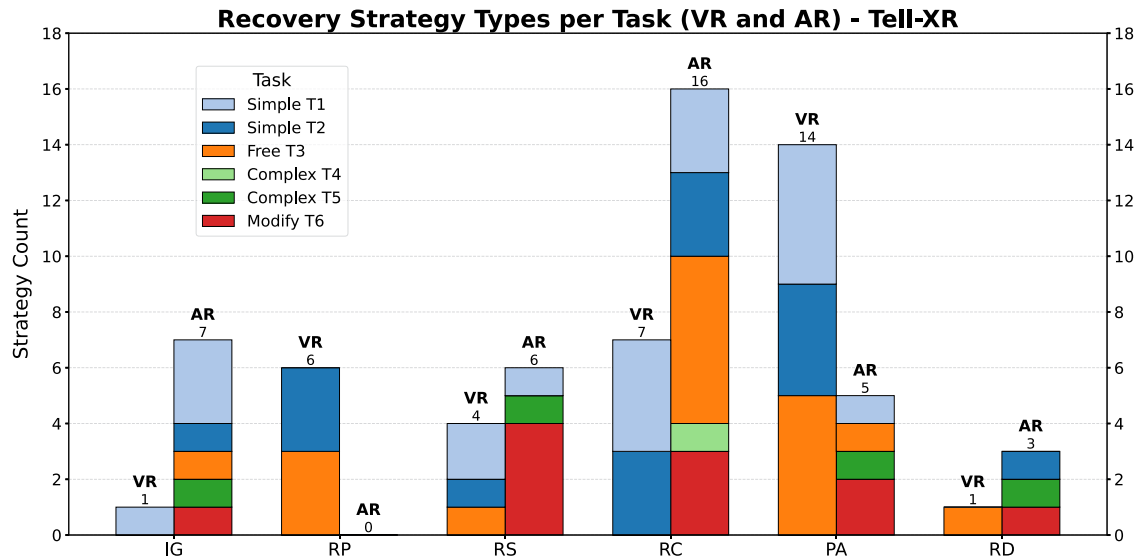


Figure 15. Distribution of user recovery strategies observed in automation authoring dialogues with Tell-XR. The six identified strategies are: ignoring the system error (IG), repeating the last intent using the same wording (RP), rephrasing the last intent using different wording (RS), failing to notice the system error and continuing the interaction unaware of it (RC), passively accepting the erroneous system input (PA), and redefining the last intent (RD). Bars report the number of occurrences in VR (left) and AR (right) settings, segmented by task type: simple automation authoring (bluish), complex automation authoring (greenish), and modification of an existing automation (red).

4.6.6. Users' recovery strategies

We also aimed to understand the strategies employed by users to recover after a chatbot error occurred during task completion. A total of 33 recovery instances in VR and 37 in AR were classified according to the following six-category taxonomy (Figure 15):

- *Aware of the error, but ignoring it (IG)*: the user realises the chatbot's error, but proceeds with the interaction (without addressing or correcting the error). This behaviour often occurred in the case of chatbot hallucination, and users' awareness of such errors was sometimes explicitly signalled aloud during the test. Figure 16 illustrates an AR case in which the user neglects an error produced by the chatbot and continues the interaction without addressing the error. In VR, we observed only a single instance of such a pattern, more precisely during T4, where participants were asked to create an automation that unlock hidden information when the visitor places a particular historical object, a sceptre, into a socket. Figure 17 reports such an instance.

- *Repetition of the last message (RP)*: the user repeats the previous utterance, trying to get a correct response through reiteration.
- *Rephrasing with the same intent (RS)*: the user reformulates the previous message using different wording while maintaining the same meaning.
- *Respond/Continue-as-if-Correct (RC)*: the user fails to notice the chatbot's error and continues the interaction, unaware of the chatbot's error.
- *Passive acceptance (PA)*: the user accepts the chatbot's erroneous input, following along with the proposed (incorrect) path without, e.g. correcting it.
- *Rephrasing different meaning (RD)*: the user reformulates the message with a modified request/meaning, thereby adapting the goal in response to the chatbot's failure.

In Figure 15, we summarise the observed strategy patterns across tasks. In the VR museum, the case in which users passively accepted what the chatbot proposed (PA, $n = 14$ or $\sim 42\%$) was the most frequent strategy: this occurred in 9 cases of chatbot wrong responses and 5 cases of chatbot hallucinations. Moving

[AR2]: Can you help me with my plants when it is too hot and I am not at home?
 [Tell-XR]: Of course! We can set up an automation that activates plant irrigation when the temperature exceeds a certain threshold and you are not at home.
 [AR2]: I would like you to open the windows to let fresh air in from outside.

Figure 16. Ignore recovery strategy Example 1 (Tell-XR study, AR setting).

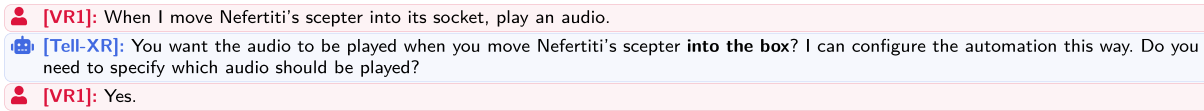


Figure 17. Ignore recovery strategy Example 2 (Tell-XR study, VR setting).

on without noticing the error was the second most common strategy (RC, $n = 7$ or $\sim 21\%$). These two approaches suggest that users tend to rely entirely on the chatbot, raising concerns about this trend, especially in a VR scenario where object behaviours may deviate from the real world. Better domain understanding could mitigate this trend. It is also important to consider that several acceptances ($n = 5$) occurred after the chatbot's preceding error propagation. This may indicate that the user accepted the chatbot's output after being unable to correct it, suggesting that the chatbot still needs further refinement to increase its flexibility and resilience to user corrections.

We observed 6 occurrences ($\sim 18\%$) where the participants repeated what was said in the previous message (RP). Following this, in 4 instances, users rephrased their message without changing its content (RS, $\sim 12\%$). These two approaches suggest that users who noticed the chatbot's error attempted to correct it in two different ways: some by reinforcing the initial input, while others rephrased their request, perhaps assuming their initial request lacked clarity. Finally, we observed one instance in both cases where the user ignored the chatbot's error and moved on, then rephrased their request, changing the underlying concept. These were the least frequent categories (IG and RD, each at $\sim 3\%$). There was a clear predominance of cases where chatbot errors either were passively accepted or went unnoticed by users, highlighting potential challenges in interaction transparency.

In the AR home, the most frequent recovery strategy was RC (Respond/Continue-as-if-Correct) ($n = 16$, $\sim 48\%$), in which users proceeded with the interaction without realising that the chatbot had made an error: this indicates that a good part of system failures went undetected by participants.

The IG case represented the second most common case ($n = 7$, $\sim 21\%$): users seem to realise the chatbot's error but they neglected this and proceeded with the interaction. This suggests a more 'pragmatic' orientation toward task completion (despite acknowledged system failures). RS (rephrasing with the same intent) had 6 instances ($\sim 18\%$), representing situations in which users reformulated their input while maintaining the original meaning: this indicates appropriate error recognition and constructive recovery behaviour on the user's side. PA (Passive Acceptance) was observed 5 times ($\sim 15\%$),

where the user accepted the chatbot's suggestion, even though the previous intent was likely different: this could raise concerns about user over-reliance on system accuracy. RD (Rephrasing – different meaning), involving reformulation with modification, occurred in 3 instances ($\sim 9\%$), while RP (Repeat-as-Is), where users repeated their previous message, did not occur at all. The predominance of RC – Respond/Continue-as-if-Correct case suggests that chatbot errors often remained unnoticed by users. Combined with IG case and PA, the majority of recovery instances involved either unawareness, conscious disregard, or passive acceptance of errors.

4.6.7. Transition among phases

The introduction of the chatbot prototype partially reshaped the dynamics of dialogue transitions across the five identified phases. Figure 18 summarises the transition frequencies among the phases, considering a total of 447 transitions in VR and 673 in AR. Overall, the trajectory still follows the sequence *Define* → *Explore* → *Refine* → *Confirm* → *Export*. The final phase continues to absorb most dialogue paths, with only a few follow-up loops or returns to *Define*.

The first notable difference between the dialogue annotations from the WoZ study and those produced with Tell-XR concerns the overall distribution of phases. The number of turns in the *Refine* phase increased in both settings, reaching nearly half of all turns in the VR Museum. With Tell-XR, 28.6% of transitions in VR and 22.9% in AR correspond to *Refine* loops. We observed several differences in both the distribution of transitions and the persistence within phases when compared to the WoZ study, in which a researcher enacted the role of the intelligent agent (see Section 3). The reason for this increased presence of the refine phase is a set of questions the chatbot systematically asks to ensure the automation definition is complete. The first is asking for additional conditions to trigger the rule: once the chatbot identifies the values of the different parameters and objects, it always asks whether the user wants to further restrict the triggering. This adds a refinement turn before asking for confirmation of the automation. Such exchanges are similar to the one reported in Figure 19.

The second reason is the bot's tendency to ask for information about the specific objects involved in the

Transition Frequencies - Tell-XR

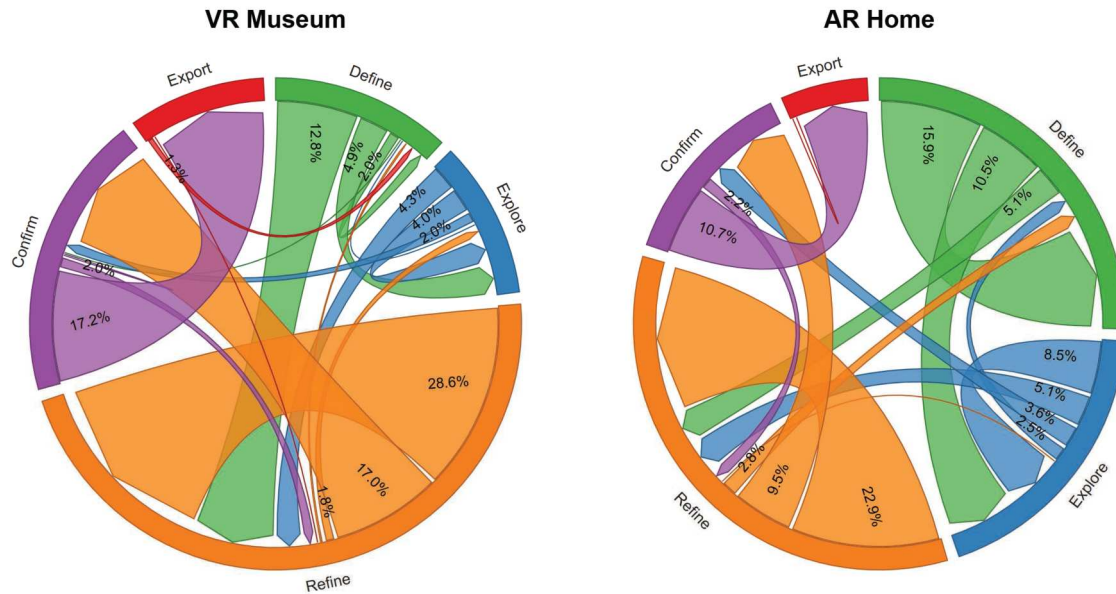


Figure 18. Chord Diagram representing transitions among the identified stages (Define, Explore, Refine, Confirm, and Export) in the collected conversations during the chatbot prototype study in the VR museum (left) and AR home (right).

automation before clarifying the user's high-level intent. For instance, participant VR3 started the definition of the automation of her choice with a cryptic intent as reported in Figure 20. The bot focussed first on identifying the object involved rather than exploring exactly what the participant meant.

The *Define* phase also grew in prominence in both settings, with transitions that did not occur in the WoZ study, including the *Define* loop in the AR home that is the dominant transition (15.9%), and the forward jump *Define* → *Refine* jump that bypasses *Explore* in the VR museum (12.8%). A possible explanation could be that, in AR, users are immersed in a physical environment (which is also quite familiar to them, e.g. the smart home). This could create a sort of 'feedback loop' between the physical space and the formulation of the goal: users may repeatedly revisit and adjust their high-level goals as they notice new services, objects, opportunities or constraints in their context, thereby leading to iterative goal refinements. Contrarily, in VR, there are no real-world anchors; thus, this reduced environmental information could make users commit to high-level goals more quickly (than in AR settings). Combining this with the higher cognitive demands required in VR settings, this could encourage

users to fixate on their early goals, thereby reducing the need for more exploratory behaviour.

In AR, it happened often that the participants started with rather abstract intents, such as just asking to create an automation, avoiding including any information about the automation itself at first or just mentioning they wanted to do something to increase the comfort, security, or energy saving in the home. This required Tell-XR to spend further turns isolating even a vague idea of the automation.

In the VR museum, the *Define* → *Refine* occurs when participants give a precise definition of the automation they want from the first turn, which results in a quick exchange to better specify some details before confirmation. Dialogues having such a transition are usually short and occur after the participant has familiarised themselves with the environment and the objects. An instance is reported in Figure 21.

The increased frequency of both *Define* and *Refine* corresponds to a contraction of turns spent in *Explore*. Although the forward transition to *Refine* and the *Explore* loop remain the most common moves within this phase, their frequencies decrease to 4.3% and 4.0% in VR, and to 8.5% and 5.1% in AR, respectively.

[Tell-XR]: How long do you want the light bulb to stay on after detecting motion?
 [AR2]: As long as there are people in the room.
 [Tell-XR]: Do you want to add a condition to turn the light bulb off when there's no more motion?

Figure 19. Example of *refine* loop due to the chatbot asking for additional conditions.

 [VR3]: Add the automation "Show in full screen"

 [Tell-XR]: Which screen do you want to show in full screen? The available options are: Nefertiti screen, Beethoven, or Anubis?

Figure 20. Example of *refine* loop due to the chatbot inquiring regarding the objects involved.

Backward transitions are even rarer than in the WoZ study, typically occurring towards the immediately preceding phase with frequencies ranging from 1.8% (*Refine* → *Explore* in VR) to 2.5% (*Explore* → *Define* in AR). We also observed a similarly infrequent backward transition from *Refine* to *Define*. The observed contraction of the *Explore* phase, combined with increased activity in *Define* and *Refine*, might suggest that users in both VR and AR settings developed more goal-oriented strategies compared to the WoZ study, thereby dedicating less time to open-ended exploration. Also, the near-absence of backward transitions (e.g. from *Refine* to earlier phases) might indicate that users rarely felt the need to revisit prior decisions: this may be due to either increased confidence in initial goal formulation or the (cognitive) cost of backtracking in immersive environments, which might have discouraged iterative reconsideration and then preferred forward progression.

The *Confirm* and *Export* phases, along with their associated transitions, did not exhibit major changes in their function.

A difference that seems to emerge between the two settings is that AR users showed frequent *Define* loops (15.9% of transitions), possibly due to a 'feedback' loop between the physical environment and goal formulation. Conversely, VR users showed a tendency to bypass exploration, with *Define* → *Refine* jumps having 12.8% of transitions, suggesting earlier goal fixation. Backward transitions remained rare across both conditions (1.8%–2.5%), which could indicate either increased confidence in initial decisions and therefore user satisfaction, or possible barriers associated with 'backtracking' in immersive environments.

4.6.8. Dialogue patterns

Together with the analysis of the transition distribution between phases, we repeated the analysis on the path described by the dialogues supported by Tell-XR, in order to identify repeating patterns in the sequences

and compare them to those we observed in the WoZ study. Similarly to the previous analysis, we counted the occurrences of the patterns we identified for each condition (VR and AR) and distinguished among dialogues for authoring a *Simple* or *Complex* automation, to *Modify* an existing one or to author an automation without any providing the participant with a specific goal (*Free*).

Figure 22 summarises the distribution of the patterns we identified. We checked the occurrence of the patterns identified in the WoZ study, together with new patterns that emerged from the interaction with Tell-XR. The four patterns previously identified in the WoZ study are reported on the left of the vertical separator, showing that they also occurred in the prototype study. The three patterns on the right, *No-E*, *Re-Define*, and *D-Loop*, instead emerged only in the prototype study.

The **Waterfall** pattern still dominates most of the authoring dialogues both in AR and VR. In the VR museum scenario, the pattern's frequency increased with respect to the previous experiment, whereas in AR, it remained stable. This confirms that the overall organisation of the dialogue in the *Define* → *Explore* → *Refine* → *Confirm* → *Export* phases is also valid when we use an LLM-based agent to guide the user in the authoring task.

The **ER-Loop** pattern, which highlights loops between the exploration and refinement phases, occurred in a very few instances this time, also in the VR museum setting. As we already pointed out, discussing the overall distribution of transitions, participants in the VR condition exhibited a similar goal-oriented strategy in defining the automation and appeared to rely less on the chatbot for exploring the environment, compared to the WoZ simulation of virtual content in the WoZ study.

The **Post-Confirm** pattern maintains instead a frequency similar to WoZ study. The task T3, which required participants to create a self-defined

 [VR11]: When the visitor clicks the yellow button under Beethoven's bust, the Ninth Symphony will appear on the screen with audio and video.

 [Tell-XR]: You want the Ninth Symphony to appear on Beethoven's screen with audio and video when the visitor clicks the yellow button. Do you need a specific condition for this automation? For example, do you want it to only happen if the visitor is in a certain position?

 [VR11]: The visitor must have already enabled media.

 [Tell-XR]: The automation will therefore be configured as follows: when the visitor clicks the yellow button, if media is enabled, the Ninth Symphony will appear on Beethoven's screen with audio and video. Can I proceed with this configuration?

Figure 21. Example of a jump transition from *define* to *refine* due to a very precise intent definition.

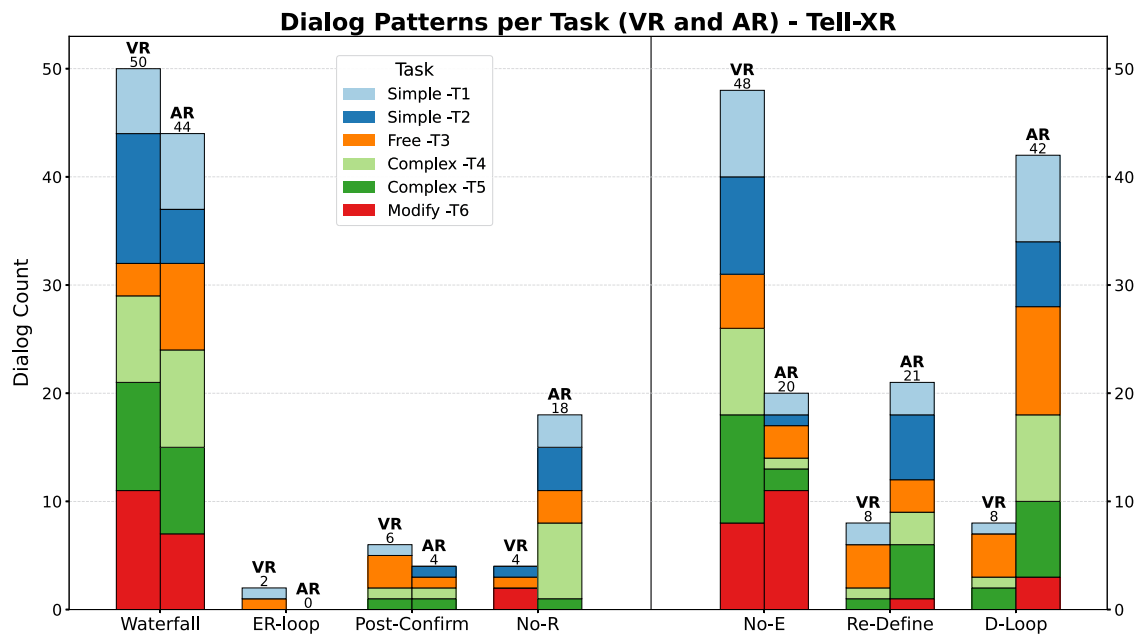


Figure 22. Distribution of automation authoring patterns in the Tell-XR. The figure reports seven patterns identified in the prototype study. The first four patterns (Waterfall, ER-Loop, Post-Confirm, No-R) were originally observed in the WoZ study and were subsequently confirmed in the prototype study, whereas the remaining three patterns (No-E, Re-Define, D-Loop) emerged only in the prototype study. A vertical separator visually distinguishes patterns previously identified (left) from those newly observed (right); however, all bars shown in the figure are derived *exclusively* from prototype study data. Bars report the number of occurrences in the VR (left) and AR (right) conditions, segmented by task type: simple automation authoring (bluish), complex automation authoring (greenish), and modification of an existing automation (red).

automation (*Free -T3* segment in Figure 22) increased the VR count.

There are no differences also in the overall frequency of dialogues that did not require any refinement phase (**No-R**). The VR museum dialogues are now aligned with the AR home in length, and this pattern usually occurred when participants provided an almost-complete automation definition since the first turn in both settings.

Besides the patterns we identified in the WoZ study, the analysis of the evaluation study dialogues highlighted **three new patterns**.

The first is the **No-E** pattern, where the dialogue does not contain any explorative phase. Such dialogues are usually short (about 7 turns in total), so the participant provides clear indications about the automation from the beginning. In addition, Tell-XR

usually tries to first clarify and finalise the partially defined rules parts rather than exploring options for those that are completely missing. All of this contributes to skipping the *Explore* phase. In the VR museum, we observed a higher number of dialogues following this pattern, with instances across all task types. In AR, by contrast, the pattern mainly occurred with automation modification tasks, as exemplified in Figures 23 and 24.

The second pattern we identified in Tell-XR dialogues is the **Re-Define**. It includes dialogues that, at some point, returned to the *Define* phase. They occurred more frequently in the AR home for both *Simple* and *Complex* tasks. Instead, in the VR museum, they appear more in *Free* tasks. The pattern indicates a change of mind in defining the automation goal midway through the task, as shown in Figures 25 and 26.

[Tell-XR]: Hi! I'm here to help you create and manage automations in your virtual scene. If you need help, let me know what you want to do!

[VR8]: When I press the green button, display the text description under the spotlight of the "The Great Wave" painting.

[Tell-XR]: Okay, do you want the description of the "The Great Wave" painting to be displayed under the spotlight when I press the green button? Do you want to add conditions, such as whether you're close to the painting or whether the spotlight is on?

[VR8]: No, that's fine.

[Tell-XR]: The automation you defined is "When I press the green button, display the text description under the spotlight of the "The Great Wave" painting." Do you want to save and export the automation?

[VR8]: Yes

Figure 23. No-E pattern Example 1 (Tell-XR study, VR setting).

[AR4]: Hey bot, I'd like to change the automation for my front door lighting.

[Tell-XR]: The automation turns on the front door light when you're within 5 km of your home. Do you want to change anything?

[AR4]: Yes, I would like to.

[Tell-XR]: What changes would you like to make to the automation? Do you want to add conditions or change the triggers?

[AR4]: Power On

[Tell-XR]: Do you want to change the time or condition for turning on the light?

[AR4]: I'd like the light to turn on when you're within 1 km of your home.

[Tell-XR]: Do you want to update the automation to turn on the light when you're within 1 km of your home?

[AR4]: Done!

Figure 24. No-E pattern Example 2 (Tell-XR study, AR setting).

[Tell-XR]: The automation defined so far is that when the motion sensor in the living room detects motion, the light in the entryway turns on.

[AR4]: Hey Bot, can you help me create a kitchen security automation?

[Tell-XR]: Sure! Do you want to activate an alarm when the smoke or gas detector in the kitchen detects a problem?

[AR4]: Yes, please.

Figure 25. Re-Define pattern Example 1 (Tell-XR study, AR setting).

[VR5]: Hey bot, let's add a sign next to the green button that says click me.

[Tell-XR]: I can't add a sign directly. Do you want to create an action for the green button that displays a message when it's pressed?

[VR5]: Ok... then... Hey bot, when a visitor enters the room, play some relaxing music.

[Tell-XR]: Do you want the relaxing music to play on one of the available speakers, such as the Nefertiti, Beethoven, Anubis, or speaker 1 or 2?

Figure 26. Re-Define pattern Example 2 (Tell-XR study, VR setting).

[Tell-XR]: Hi! I can help you create automations for your home. Do you want to improve security, comfort, or energy savings?

[AR9]: Can you give me some tips for improving comfort?

[Tell-XR]: You can turn on the light bulb when the motion sensor detects presence. Do you want to continue with this automation?

[AR9]: Or?

[Tell-XR]: You can activate the air purifier when the air quality is low. Do you want to set up this automation?

[AR9]: Hmm... okay!

[Tell-XR]: [continues refining the parameters for the automation]

Figure 27. D-Loop pattern Example 1 (Tell-XR study, AR setting).

The last pattern we identified is looping over the *Define* phase (**D-Loop**), indicating turns spent in revisiting and adjusting the high-level goal of the automation. Such a pattern occurs often in AR, as visible in Figure 22, and the dialogues under this pattern are characterised by participants who seek the bot's guidance in clarifying or suggesting ideas for the automation, as highlighted in Figure 27.

Compared to the WoZ study, the decrease in ER-Loop patterns, especially in the VR condition, can indicate that users engaged in less iterative cycling between exploration and refinement phases. This shift toward more goal-oriented, linear interaction strategies suggests that the chatbot facilitated more efficient task completion; however, this efficiency may come at the cost of reduced discovery of alternative automation possibilities.

5. Discussion

Building on the formative WoZ insights, Tell-XR operationalised a multi-agent conversational workflow across the identified phases, along with a context-aware routing. The evaluation across a VR museum and an AR smart home shows that the chatbot engineered around the identified phases effectively supported participants in the automation authoring task. The analysis of the collected data revealed the occurrence of patterns already identified during the WoZ study, together with other ones resulting from the interaction with a real system rather than a simulated agent. In addition, we analysed the errors and the recovery strategies while using the Tell-XR prototype. The common traits and identified differences provide useful insights into how users articulate their intent when creating automations using a working chatbot prototype.

5.1. RQ1: dialogue structure

The results presented in Section 4.6.7 address RQ1 by identifying a structured organisation of conversational automation authoring into five phases: Define, Explore, Refine, Confirm, and Export. These phases were first derived from the Wizard-of-Oz study and subsequently embedded into the design of the Tell-XR prototype. The prototype study provides evidence that this phase-based structure remains applicable and observable when operationalised in a working system. The analysis of transitions further shows that the authoring process is not linear but includes iterative loops, particularly in the Explore and Refine phases.

Despite differences in interaction paradigms and devices (HMDs versus smartphones), users adopted similar strategies for rule definition in both VR and AR settings. The studies show that users adapt rule formulation to the specific affordances of each environment. Moreover, differences in user goals, ranging from entertainment-oriented scenarios (museum) to utility-driven ones (smart home), did not affect the overall strategy for defining automations.

While the overall structure of the dialogue remains consistent across all settings and experiments, we found some differences that can inform the design and implementation of similar agents. First of all, Tell-XR's systematic follow-ups for completing all rule parts, including asking for additional conditions or for more information about objects and parameters, led to an increase in turns spent in refining the currently available definition rather than exploring other options, which instead was the main activity when a human interpreted the intelligent agent. Other differences included direct jumps between *Define* and *Refine* in the VR museum, which indicated a precise, goal-oriented specification of the automation, and the occurrence of *Define* loops in the AR smart home, which usually included requests for guidance to the intelligent agent in getting possible automations.

Overall, these observations highlight a shift from a more exploratory approach in the WoZ setting (when users talked to human agents), towards a more goal-oriented conversation with the LLM-based prototype: when talking to Tell-XR, participants sometimes asked the agent for ready-to-use options for entire automations or some parts (while this did not happen with a human agent). Also, especially in the unfamiliar VR domain, they tended to over-rely on the system and accept its (sometimes erroneous) suggestions. While we interpret this as a tendency to over-reliance (Lai and Tan 2019) on AI tools, this suggests that we need to further investigate the role of the chatbot as an

automation *recommender*, in addition to its ability to support authoring. In this perspective, AI-based automation generation can be seen as an extension of recommendation systems, where candidate automations are dynamically tailored to the user's context and intent, while the end user retains an active role in evaluating and validating the resulting behaviour.

5.2. RQ2: dialogue patterns

RQ2 is addressed by identifying recurring dialogue patterns that capture how users traverse the conversational phases. Patterns such as Waterfall, ER-loop, Post-Confirm, and No-R reveal distinct strategies ranging from linear, goal-driven interactions to more exploratory and iterative behaviours. In particular, we noted the persistence of a *Waterfall* pattern, the usual flow when users start with a clear automation goal.

The distribution of these patterns varies across conditions. In the WoZ VR setting, exploratory loops occurred more frequently, suggesting that participants used the conversational interaction not only to define automations, but also to discover the environment and the possible actions with the system. By contrast, the LLM-based prototype produced more linear and concise dialogues, where users appeared to adopt a more goal-oriented strategy. This shift may indicate that the chatbot reduced the conversational overhead required to complete tasks by guiding users more directly toward viable automation definitions. However, the reduction in exploratory loops could be considered as a design trade-off: while interactions became more efficient, users may have engaged less with alternative automation opportunities or creative exploration of the environment; similar trade-offs have been also reported in recent work on human-AI co-creation (Wen et al. 2025). These findings show that conversational authoring is not only structured at the phase level, but also characterised by higher-level interaction strategies that reflect users' goals and the characteristics of the target domain.

5.3. RQ3: error types and recovery strategies

RQ3 is addressed through the combined analysis of error types and user recovery behaviours. The results show that errors originate from multiple sources, including users, the chatbot, and speech recognition; while analysing the strategies users adopted to recover from errors, we identified common behaviours and differences between the two settings, which we describe in the following.

On the one hand, we observed in both scenarios (AR/VR) that the combined prevalence of cases in which participants were aware of the error but ignored it (IG), failed to notice it (RC), or passively accepted the erroneous chatbot's output (PA) was the predominant tendency, with none of them involving active repair. In particular, Passive Acceptance was the most common strategy in the VR (museum) scenario, while RC (Respond/Continue-as-if-Correct) was more prevalent in the AR (smart-home) scenario. Although both these patterns leave chatbot errors uncorrected, they could have derived from opposite sources: reliance on the system in VR, confidence in the users' own knowledge in AR. In VR, an unfamiliar domain where the lack of physical constraints widens what could be automated, participants had little prior knowledge/basis for independently evaluating the chatbot's output, so they tended to rely on it to discover possibilities and tended to accept its suggestions. Thus, in this setting, component-level confidence indicators (for a rule's event, condition, and action), derived from structural validation, can flag uncertain parts and prompt the over-trusting users we observed in VR to look more closely. In AR, by contrast, users' familiarity with real-world objects and their properties may have reduced their attention to whether the chatbot had properly translated their original intent, so its errors went more unnoticed (RC): in this setting, chatbots could need mechanisms that make potential errors visible rather than relying on users to catch them. A dedicated verification ('judge') agent that checks the generated rule against the preceding dialogue can surface mismatches between what users asked for and what the system actually produced, catching errors that confident, domain-familiar users in AR may not notice.

On the other hand, the relatively low frequency of more 'active' recovery strategies (cases RP: users repeating the previous message; RS: users rephrasing, keeping the same meaning; and RD: users rephrasing, using a different concept) suggests that users either may want to avoid the burden to correct system failures actively, or they could miss the motivation to do that (e.g. they are less inclined to think that a reformulation can actually help the system improve its outcome). In VR, users mainly repeated the initial input (RP), whereas in AR participants preferred a rephrasing strategy that retained the same concept (RS). However, this difference may also be attributed to the higher number of voice recognition errors observed in VR compared to the AR scenario; these failures also led to different chatbot errors and hallucinations, often driving users to repeat their previous message when the chatbot's response made it evident that a failure in the voice recognition system had occurred.

As a design implication, future iterations should consider facilitating error recovery not only by lowering the effort required for reformulation (e.g. by offering alternative rephrasing and requesting confirmation), but also by integrating a recommendation agent that leverages interaction data from other users. Prior work on recommendation-supported rule authoring (Mattioli and Paternò 2025) shows that presenting suggestions incrementally at the level of single rule elements can increase user confidence, reduce ambiguity, and prevent common modelling errors, particularly in complex automations. In our context, such a recommendation agent could propose reformulations, intents, or rule parts that other users have successfully adopted in similar situations, thereby providing concrete, context-sensitive guidance. Beyond improving efficiency, recommendations could encourage more active recovery strategies by making users aware of alternative formulations that are meaningful and actionable, rather than LLM-generated. Moreover, adaptive recommendation mechanisms that account for users' authoring strategies and preferences could help mitigate cascading errors by steering conversations back onto a valid trajectory, especially when the chatbot's responses leave the user with no way to resume the conversation after an error. Explicitly communicating that recommendations are derived from collective usage patterns may further support trust, positioning the chatbot as a system that can make mistakes but can also improve over time and encouraging users to think critically and reflect on its suggestions.

5.4. Limitations

A number of limitations should be considered when interpreting the results of this work. While the study provides insights into conversational automation authoring in XR environments, several aspects of the experimental design and evaluation context may limit the generalisability of the findings. First, although the WoZ study and the Tell-XR prototype study followed the same experimental structure and tasks, the two setups relied on fundamentally different interaction mechanisms. In the WoZ condition, participants interacted with a human-mediated conversational system, whereas the prototype relied on an LLM-based architecture with predefined prompting and routing strategies. As a result, similarities and differences in the observed dialogue patterns should be interpreted qualitatively rather than as directly comparable measurements across studies. In particular, some interaction dynamics may have been influenced by the conversational behaviour of the underlying system itself, including its guidance

strategies, interpretation capabilities, and conversational constraints.

Second, the ecological validity of the study is limited by the experimental nature of the evaluated environments. Both the VR museum and the AR smart-home scenarios were specifically designed for the study and exposed participants to a constrained set of tasks, devices, and interaction possibilities. As a result, the identified dialogue patterns may not fully generalise to real-world XR settings, where environments are typically wider, less predictable, and socially situated. The study captures only first-use interactions and does not include a longitudinal evaluation. Several behaviours observed during the experiments, particularly in the VR condition, appear strongly connected to participants progressively learning the environment and discovering available affordances. As users become more familiar with the system over time, the identified dialogue patterns and transition dynamics may evolve substantially. Future work should therefore investigate how conversational automation authoring changes with prolonged use, repeated interactions, and increasing user expertise.

Another limitation concerns the composition of the participant samples across the AR and VR studies. Although both groups belonged to the same target population of non-programmers, they differed in age range and background, and participants were recruited through convenience sampling within an Italian context. Consequently, some of the behavioural differences observed between the AR and VR conditions may partially depend on sample characteristics rather than solely on the interaction setting itself. In addition, the relatively limited and culturally homogeneous sample may restrict the generalisability of the findings to users with different linguistic, cultural, or educational backgrounds. For this reason, the comparisons between AR and VR should be interpreted primarily as qualitative observations aimed at identifying interaction patterns, strategies, and barriers, rather than as definitive evidence of environment-specific effects.

Finally, the observed indications of overreliance and trust toward the bot were inferred from participants' behaviour and dialogue log analysis, rather than being directly assessed through validated trust-in-automation measurement instruments, which we plan to perform as future work.

6. Conclusion and future work

This paper presents an analysis of the structure and patterns in conversations between end users and an intelligent chatbot for creating automation rules. We collected data from two user studies: a Wizard of Oz study and a

second study involving a functional chatbot prototype. Both studies took place in two experimental settings—a virtual museum in VR and a smart home in AR (Mobile). Our analysis of the conversational data revealed consistent aspects across both interaction settings (AR vs. VR) and domains (museum vs. smart home), including the dialogue structure (*Define* → *Explore* → *Refine* → *Confirm* → *Export*) and recurring patterns in how participants described automations to the intelligent agent.

We examined both system- and user-generated errors, considering their individual occurrences and cascading effects, as well as strategies users employed to handle them. The results indicate that system errors, particularly in VR, are more likely to propagate and that users often struggle to detect such errors. Participants also tended to over-rely on the system's *perceived* accuracy, highlighting the need for better support to foster critical engagement with LLM-enhanced conversational automation tools.

Based on these findings, future work will focus on enhancing the chatbot architecture to include components that prevent or mitigate the impact of system errors on task progression. Another key direction is conducting longitudinal studies to understand how user strategies, dialogue patterns, error types, and repair behaviours evolve with increasing, together with techniques for fostering an appropriate reliance on the chatbot suggestions. Finally, while this work concentrated on end-user interaction with a conversational agent in immersive XR environments, a systematic comparison with traditional, non-conversational authoring tools is needed to clarify the role of conversational systems within the broader landscape of XR end-user development tools.

Notes

1. Technical details on how the prototype works are briefly discussed in this paper, they are available in Carcangiu et al. (2026) together with information on the usability evaluation.
2. In the analysis of the phase transitions, we also included the dialogues resulting from unsuccessfully completed tasks. In these cases, the flow ends with the last label assigned to the last turn.
3. <https://www.home-assistant.io>

Author contributions

CRedit: **Alessandro Carcangiu**: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing – original draft, Writing – review & editing; **Marco Manca**: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project

administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing; **Jacopo Mereu**: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing – original draft, Writing – review & editing; **Carmen Santoro**: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing; **Ludovica Simeoli**: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing – original draft, Writing – review & editing; **Lucio Davide Spano**: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

Funded by the Italian PRIN 2022 ‘EUD4XR: End-User Development for eXtended Reality’ funded by the Italian MUR and European Union - NextGenerationEU, Mission 4, Component 2, Investment 1.1 (grant F53D23004380006, MUR code 2022Y3CZZT) <https://prin.unica.it/eud4xr/>. Jacopo Mereu participated in this research while attending the PhD programme in Mathematics and Computer Science at the University of Cagliari (39th cycle), supported by a scholarship funded under D.M. n. 118 (2.3.2023), within the Italian National Recovery and Resilience Plan (PNRR) – funded by the European Union – NextGenerationEU – Mission 4, Component 1, Investment 4.1.

Declaration on AI usage

During the writing of this paper, the author(s) used Grammarly Pro and ChatGPT to fix grammatical errors, rephrase and improve text quality. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the published article.

ORCID

Alessandro Carcangiu  <https://orcid.org/0009-0007-8989-3384>

Marco Manca  <https://orcid.org/0000-0003-1029-9934>

Jacopo Mereu  <https://orcid.org/0009-0008-7521-7876>

Carmen Santoro  <https://orcid.org/0000-0002-0556-7538>

Ludovica Simeoli  <https://orcid.org/0009-0009-3897-6520>

Lucio Davide Spano  <http://orcid.org/0000-0001-7106-0463>

References

Adão, T., L. Pádua, M. Fonseca, L. Agrellos, J. J. Sousa, L. Magalhães, and E. Peres. 2018. “A Rapid Prototyping

Tool to Produce 360 Video-Based Immersive Experiences Enhanced with Virtual/multimedia Elements.” *Procedia Computer Science* 138:441–453. <https://doi.org/10.1016/j.procs.2018.10.062>.

Adler, R. F., F. Iacobelli, and Y. Gutstein. 2016. “Are You Convinced? a Wizard of Oz Study to Test Emotional vs. Rational Persuasion Strategies in Dialogues.” *Computers in Human Behavior* 57:75–81. <https://doi.org/10.1016/j.chb.2015.12.011>.

Aghel Manesh, S., T. Zhang, Y. Onishi, K. Hara, S. Bateman, J. Li, and A. Tang. 2024. “How People Prompt Generative AI to Create Interactive VR Scenes.” In *Proceedings of the 2024 ACM Designing Interactive Systems Conference, DIS '24*, 2319–2340. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3643834.3661547>.

Andrao, M., F. Gini, A. Bucchiarone, A. Marconi, B. Treccani, and M. Zancanaro. 2023. “Enhance Gamification Design through End-User Development: A Proposal.” In *CEUR Workshop Proceedings*. Vol. 3408. IS-EUD 2023 Workshops Cagliari, Italy.

Ariano, R., M. Manca, F. Paternò, and C. Santoro. 2023. “Smartphone-Based Augmented Reality for End-User Creation of Home Automations.” *Behaviour & Information Technology* 42:124–140. <https://doi.org/10.1080/0144929X.2021.2017482>.

Artizzu, V., G. Cherchi, D. Fara, V. Frau, R. Macis, L. Pitzalis, A. Tola, I. Blecic, and L. D. Spano. 2022. “Defining Configurable Virtual Reality Templates for End Users.” *Proceedings of the ACM on Human-Computer Interaction* 6: 163. <https://doi.org/10.1145/3534517>.

Ashktorab, Z., M. Jain, Q. V. Liao, and J. D. Weisz. 2019. “Resilient Chatbots: Repair Strategy Preferences for Conversational Breakdowns.” In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–12, Glasgow, Scotland, UK.

Asisof, A. 2022. “Towards a Comprehensive Repair Framework for Human-Chatbot Interaction: The Case of Rephrasing.” In *Proceedings of the 22nd ACM International Conference on Intelligent Virtual Agents*, 1–3, University of Algarve, Faro, Portugal.

Barricelli, B. R., F. Cassano, D. Fogli, and A. Piccinno. 2019. “End-User Development, End-User Programming and End-User Software Engineering: A Systematic Mapping Study.” *Journal of Systems and Software* 149:101–137. <https://doi.org/10.1016/j.jss.2018.11.041>.

Barricelli, B. R., D. Fogli, S. Gallo, D. Guizzardi, S. Maenza, A. Mattioli, and F. Paternò. 2025. “An EUD Approach to Creating Feasible and Energy-saving Automations for Smart Homes.” In *International Symposium on End User Development*, 3–21. Springer.

Bekele, M. K., R. Pierdicca, E. Frontoni, E. S. Malinverni, and J. Gain. 2018. “A Survey of Augmented, Virtual, and Mixed Reality for Cultural Heritage.” *Journal on Computing and Cultural Heritage* 11: 7. <https://doi.org/10.1145/3145534>.

Bellucci, A., A. Vianello, Y. Florack, L. Micallef, and G. Jacucci. 2019. “Augmenting Objects at Home through Programmable Sensor Tokens: A Design Journey.” *International Journal of Human-Computer Studies* 122:211–231. <https://doi.org/10.1016/j.ijhcs.2018.09.002>.

Blečić, I., S. Cuccu, F. A. Fanni, V. Frau, R. Macis, V. Saiu, M. Senis, L. D. Spano, and A. Tola. 2021. “First-Person

- Cinematographic Videogames: Game Model, Authoring Environment, and Potential for Creating Affection for Places." *Journal on Computing and Cultural Heritage* 14: 18. <https://doi.org/10.1145/3446977>.
- Brackenbury, W., A. Deora, J. Ritchey, J. Vallee, W. He, G. Wang, M. L. Littman, and B. Ur. 2019. "How Users Interpret Bugs in Trigger-Action Programming." In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–12, Glasgow, Scotland, UK.
- Bradley, J., D. Benyon, O. Mival, and N. Webb. 2010. "Wizard of Oz Experiments and Companion Dialogues." In *Proceedings of the 24th BCS Interaction Specialist Group Conference, BCS '10*, 117–123. Swindon, GBR: BCS Learning & Development Ltd.
- Braggaar, A., J. Verhagen, G. Martijn, and C. Liebrecht. 2023. "Conversational Repair Strategies to Cope with Errors and Breakdowns in Customer Service Chatbot Conversations." In *International Workshop on Chatbot Research and Design*, 23–41. Springer.
- Braun, V., and V. Clarke. 2006. "Using Thematic Analysis in Psychology." *Qualitative Research in Psychology* 3:77–101. <https://doi.org/10.1191/1478088706qp063oa>.
- Carcangiu, A., M. Manca, J. Mereu, C. Santoro, L. Simeoli, and L. D. Spano. 2025. "Conversational Rule Creation in XR: Users Strategies in VR and AR Automation." In *End-User Development*, edited by C. Santoro, A. Schmidt, M. Matera, and A. Bellucci, 59–79. Cham: Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-95452-8_4.
- Carcangiu, A., M. Manca, J. Mereu, C. Santoro, L. Simeoli, and L. D. Spano. 2026. "Tell-XR: Conversational End-User Development of XR Automations." In *Human-Computer Interaction – INTERACT 2025*, edited by C. Ardito, S. Diniz Junqueira Barbosa, T. Conte, A. Freire, I. Gasparini, P. Palanque, and R. Prates, 617–641. Cham: Springer Nature Switzerland.
- Chen, J., X. Wu, T. Lan, and B. Li. 2025. "Llmer: Crafting Interactive Extended Reality Worlds with Json Data Generated by Large Language Models." *IEEE Transactions on Visualization and Computer Graphics* 31:2715–2724. <https://doi.org/10.1109/TVCG.2025.3549549>.
- Chen, X., S. Chen, Z. Jin, H. Bian, Z. Chen, and H. Li. 2025. "Expressing the Needs in Smart Home: What is the End Users' Favorite Way." *ACM Transactions on Computer-Human Interaction* 32 (2): 16. <https://doi.org/10.1145/3715114>.
- Cheng, F., V. Zouhar, S. Arora, M. Sachan, H. Strobelt, and M. El-Assady. 2024. "Relic: Investigating Large Language Model Responses Using Self-consistency." In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, 1–18, Honolulu, HI, USA.
- Clark, M., M. W. Newman, and P. Dutta. 2017. "Devices and Data and Agents, Oh My: How Smart Home Abstractions Prime End-User Mental Models." *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1 (3): 44. <https://doi.org/10.1145/3132031>.
- Corno, F., L. De Russis, and A. Monge Roffarello. 2019. "Empowering End Users in Debugging Trigger-Action Rules." In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI '19*, 1–13. New York, NY, USA: Association for Computing Machinery.
- Corno, F., L. De Russis, and A. Monge Roffarello. 2020. "HeyTAP: Bridging the Gaps between Users' Needs and Technology in IF-THEN Rules via Conversation." In *Proceedings of the 2020 International Conference on Advanced Visual Interfaces, AVI '20*. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3399715.3399905>.
- Coutaz, J., and J. L. Crowley. 2016. "A First-Person Experience with End-User Development for Smart Homes." *IEEE Pervasive Computing* 15:26–39. <https://doi.org/10.1109/MPRV.2016.24>.
- De La Torre, F., C. M. Fang, H. Huang, A. Banburski-Fahey, J. Amores Fernandez, and J. Lanier. 2024. "LLMR: Real-Time Prompting of Interactive Worlds Using Large Language Models." In *Proceedings of the CHI Conference on Human Factors in Computing Systems, CHI '24*. New York, NY, USA: Association for Computing Machinery.
- De Russis, L., and F. Corno. 2015. "Homerules: A Tangible End-User Programming Interface for Smart Homes." In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, 2109–2114, Seoul, Republic of Korea.
- Desolda, G., C. Ardito, and M. Matera. 2017. "Empowering End Users to Customize Their Smart Environments: Model, Composition Paradigms, and Domain-Specific Tools." *ACM Transactions on Computer-Human Interaction* 24 (2): 12. <https://doi.org/10.1145/3057859>.
- Desolda, G., F. Greco, F. Guarnieri, N. Mariz, and M. Zancanaro. 2021. "SENSATION: an Authoring Tool to Support Event-state Paradigm in End-User Development." In *Human-Computer Interaction – INTERACT 2021: 18th IFIP TC 13 International Conference, Bari, Italy, August 30–September 3, 2021, Proceedings, Part II* 18, 373–382. Springer.
- Dey, A. K., and A. Newberger. 2009. "Support for Context-Aware Intelligibility and Control." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 859–868, Boston, MA, USA.
- Dippold, D. 2023. "Can I Have the Scan on Tuesday? User Repair in Interaction with a Task-Oriented Chatbot and the Question of Communication Skills for AI." *Journal of Pragmatics* 204:21–32. <https://doi.org/10.1016/j.pragma.2022.12.004>.
- Fanni, F. A., M. Senis, A. Tola, F. Murru, M. Romoli, L. D. Spano, I. Blečić, and G. A. Trunfio. 2019. "PAC-PAC: End User Development of Immersive Point and Click Games." In *End-User Development*, edited by A. Malizia, S. Valtolina, A. Morch, A. Serrano, and A. Stratton, 225–229. Cham: Springer International Publishing.
- Farias, F. M. D., D. P. De Mattos, G. Ghinea, and D. C. Muchaluat-Saade. 2022. "Immersive Authoring of 360 Degree Interactive Applications." *IEEE Access* 10:115205–115221. <https://doi.org/10.1109/ACCESS.2022.3217799>.
- Fogli, D., M. Peroni, and C. Stefani. 2017. "Imathome: Making Trigger-Action Programming Easy and Fun." *Journal of Visual Languages & Computing* 42:60–75. <https://doi.org/10.1016/j.jvlc.2017.08.003>.
- Følstad, A., and C. Taylor. 2019. "Conversational Repair in Chatbots for Customer Service: The Effect of Expressing Uncertainty and Suggesting Alternatives." In *International Workshop on Chatbot Research and Design*, 201–214. Springer.

- Frau, V., L. D. Spano, V. Artizzu, and M. Nebeling. 2023. "XRSpotlight: Example-Based Programming of XR Interactions Using a Rule-Based Approach." *Proceedings of the ACM on Human-Computer Interaction* 7: 185. <https://doi.org/10.1145/3593237>.
- Gallo, S., F. Paternò, and A. Malizia. 2024. "A Conversational Agent for Creating Automations Exploiting Large Language Models." *Personal and Ubiquitous Computing* 28:931–946. <https://doi.org/10.1007/s00779-024-01825-5>.
- Ghiani, G., M. Manca, F. Paternò, and C. Santoro. 2017. "Personalization of Context-Dependent Applications through Trigger-Action Rules." *ACM Transactions on Computer-Human Interaction* 24: 14. <https://doi.org/10.1145/3057861>.
- Giudici, M., L. Padalino, G. Paolino, I. Paratici, A. I. Pascu, and F. Garzotto. 2024. "Designing Home Automation Routines Using an LLM-based Chatbot." *Designs* 8:43. <https://doi.org/10.3390/designs8030043>.
- Gnewuch, U., and F. Reinkemeier. 2025. "Overcoming Breakdowns in Customer-Chatbot Interaction: Design and Impact of Collaborative Repair Strategies." *Management Information Systems Quarterly* 50 (2): 1–33.
- Heun, V., J. Hobin, and P. Maes. 2013. "Reality Editor: Programming Smarter Objects." In *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication*, 307–310, Zurich, Switzerland.
- Huang, J., and M. Cakmak. 2015. "Supporting Mental Model Accuracy in Trigger-Action Programming." In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 215–225, Osaka, Japan.
- Huang, T. H. K., A. Azaria, O. J. Romero, and J. P. Bigham. 2019. "Instructablecrowd: Creating If-Then Rules for Smartphones via Conversations with the Crowd." *Human Computation* 6:113–146. <https://doi.org/10.15346/hc.v6i1>.
- IFTTT. n.d.. *IFTTT: Connect Your Apps and Devices*. Accessed February 24, 2025. <https://ifttt.com/>.
- Ikeda, B., and D. Szafr. 2024. "Programar: Augmented Reality End-User Robot Programming." *The Journal of Human-Robot Interaction* 13 (1): 15.
- Izadi, S., and M. Forouzanfar. 2024. "Error Correction and Adaptation in Conversational AI: a Review of Techniques and Applications in Chatbots." *Ai* 5:803–841. <https://doi.org/10.3390/ai5020041>.
- Jennings, N., H. Wang, I. Li, J. Smith, and B. Hartmann. 2024. "What's the Game, Then? Opportunities and Challenges for Runtime Behavior Generation." In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology, UIST '24*. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3654777.3676358>.
- Kim, Y., J. Lee, S. Kim, J. Park, and J. Kim. 2024. "Understanding Users' Dissatisfaction with Chatgpt Responses: Types, Resolving Tactics, and the Effect of Knowledge Level." In *Proceedings of the 29th International Conference on Intelligent User Interfaces*, 385–404, Greenville, SC, USA.
- Knierim, P., P. W. Woźniak, Y. Abdelrahman, and A. Schmidt. 2019. "Exploring the Potential of Augmented Reality in Domestic Environments." In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services, MobileHCI '19*. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3338286.3340142>.
- Ko, A. J., B. A. Myers, and H. H. Aung. 2004. "Six Learning Barriers in End-User Programming Systems." In *2004 IEEE Symposium on Visual Languages -- Human Centric Computing*, 199–206, Rome, Italy.
- Kubitza, T., and A. Schmidt. 2015. "Towards a Toolkit for the Rapid Creation of Smart Environments." In *End-User Development: 5th International Symposium, IS-EUD 2015, Madrid, Spain, May 26-29, 2015. Proceedings* 5, 230–235. Springer.
- Lai, V., and C. Tan. 2019. "On Human Predictions with Explanations and Predictions of Machine Learning Models: A Case Study on Deception Detection." In *Proceedings of the Conference on Fairness, Accountability, and Transparency, FAT* '19*, 29–38. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3287560.3287590>.
- Lee, G. A., and G. J. Kim. 2009. "Immersive Authoring of Tangible Augmented Reality Content: A User Study." *Journal of Visual Languages & Computing* 20:61–79. <https://doi.org/10.1016/j.jvlc.2008.07.001>.
- Lee, G. A., G. J. Kim, and M. Billinghamhurst. 2005. "Immersive Authoring: What You Experience is What You Get (wyxi-wyg)." *Communications of the ACM* 48:76–81. <https://doi.org/10.1145/1070838.1070840>.
- Lee, J., F. Aleotti, D. Mazala, G. Garcia-Hernando, S. Vicente, O. J. Johnston, I. Kraus-Liang, et al. 2025. "ImagineAR: AI-Assisted in-Situ Authoring in Augmented Reality." In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology, UIST '25*. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3746059.3747635>.
- Lee, S., and J. Choi. 2017. "Enhancing User Experience with Conversational Agent for Movie Recommendation: Effects of Self-disclosure and Reciprocity." *International Journal of Human-Computer Studies* 103:95–105. <https://doi.org/10.1016/j.ijhcs.2017.02.005>.
- Leiser, F., S. Eckhardt, V. Leuthe, M. Knaeble, A. Mädche, G. Schwabe, and A. Sunyaev. 2024. "Hill: A Hallucination Identifier for Large Language Models." In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, 1–13, Honolulu, HI, USA.
- Leonardi, N., M. Manca, F. Paternò, and C. Santoro. 2019. "Trigger-Action Programming for Personalising Humanoid Robot Behaviour." In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 1–13, Glasgow, Scotland, UK.
- Liu, X., Y. Shi, C. Yu, C. Gao, T. Yang, C. Liang, and Y. Shi. 2023. "Understanding In-Situ Programming for Smart Home Automation." *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 7 (2): 66. <https://doi.org/10.1145/3596254>.
- Mahroo, A., L. Greci, and M. Sacco. 2019. "HoloHome: An Augmented Reality Framework to Manage the Smart Home." In *Augmented Reality, Virtual Reality, and Computer Graphics: 6th International Conference, AVR 2019, Santa Maria al Bagno, Italy, June 24–27, 2019, Proceedings, Part II*, 137–145. Berlin, Heidelberg: Springer-Verlag.

- Manca, M., F. Paternò, C. Santoro, and L. Corcella. 2019. "Supporting End-User Debugging of Trigger-Action Rules for Iot Applications." *International Journal of Human-Computer Studies* 123:56–69. <https://doi.org/10.1016/j.ijhcs.2018.11.005>.
- Mattioli, A., and F. Paternò. 2025. "Personalised Recommendations for Daily Automations Controlled by Mobile Augmented Reality." *Proceedings of the ACM on Human-Computer Interaction* 9 (4): 1–22. <https://doi.org/10.1145/3734863>.
- Mitchell, E., and L. Mamykina. 2021. "From the Curtain to Kansas: Conducting Wizard-of-Oz Studies in the Wild." In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems, CHI EA '21*. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3411763.3443446>.
- Monge Roffarello, A., and L. De Russis. 2023. "Defining Trigger-Action Rules via Voice: A Novel Approach for End-User Development in the IoT." In *International Symposium on End User Development*, 65–83. Springer.
- Moore, R. J., S. An, and O. H. Marrese. 2024. "Understanding Is a Two-Way Street: User-Initiated Repair on Agent Responses and Hearing in Conversational Interfaces." *Proceedings of the ACM on Human-Computer Interaction* 8:1–26.
- Mugunthan, T., and S. Gibbons. 2024. *The 4 Stages of AI Image Generation*, Nielsen Norman Group Blog. Accessed February 7, 2025. <https://www.nngroup.com/articles/ai-imagegen-stages/>.
- Noura, M., S. Heil, and M. Gaedke. 2020. *Natural Language Goal Understanding for Smart Home Environments*, in *Proceedings of the 10th International Conference on the Internet of Things, IoT '20*, New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3410992.3410996>.
- OpenAI. 2024. "GPT-4o System Card." arXiv preprint arXiv:2410.21276, <https://arxiv.org/abs/2410.21276>.
- Paternò, F., and C. Santoro. 2019. "End-User Development for Personalizing Applications, Things, and Robots." *International Journal of Human-Computer Studies* 131:120–130. <https://doi.org/10.1016/j.ijhcs.2019.06.002>.
- Seiger, R., M. Gohlke, and U. Aßmann. 2019. "Augmented Reality-Based Process Modelling for the Internet of Things with Holoflows." In *Enterprise, Business-Process and Information Systems Modeling: 20th International Conference, BPMDS 2019, 24th International Conference, EMMSAD 2019, Held at CAiSE 2019, Rome, Italy, June 3–4, 2019, Proceedings 20*, 115–129. Springer.
- Sharma, N., Z. Zhang, D. Lee, N. Krishnan, G. J. Ren, Z. Xiao, and Y. Li. 2026. "Feedback by Design: Understanding and Overcoming User Feedback Barriers in Conversational Agents." In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems*, 1–22, Barcelona, Spain.
- Steed, A., and M. Slater. 1996. "A Dataflow Representation for Defining Behaviours within Virtual Environments." In *Proceedings of the IEEE 1996 Virtual Reality Annual International Symposium*, 163–167, Santa Clara, CA, USA.
- Stefanidi, E., D. Arampatzis, A. Leonidis, M. Korozi, M. Antona, and G. Papagiannakis. 2020. "Magiplay: An Augmented Reality Serious Game Allowing Children to Program Intelligent Environments." *Transactions on Computational Science XXXVII: Special Issue on Computer Graphics* 12230: 144–169. <https://doi.org/10.1007/978-3-662-61983-4>.
- Stefanidi, E., D. Arampatzis, A. Leonidis, and G. Papagiannakis. 2019. "BricklAyeR: A Platform for Building Rules for AmI Environments in AR." In *Advances in Computer Graphics*, edited by M. Gavrilova, J. Chang, N.M. Thalmann, E. Hitzler, and H. Ishikawa, 417–423. Cham: Springer International Publishing.
- Sun, Y., A. Armengol-Urpi, S. N. Reddy Kantareddy, J. Siegel, and S. Sarma. 2019. "MagicHand: Interact with IoT Devices in Augmented Reality Environment." In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, 1738–1743, Osaka, Japan.
- Takala, T. M. 2014. "RUIS: A Toolkit for Developing Virtual Reality Applications with Spatial Interaction." In *Proceedings of the 2nd ACM Symposium on Spatial User Interaction, SUI '14*, 94–103. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/2659766.2659774>.
- Timpe-Laughlin, V., R. Divekar, T. Sydorenko, J. Dombi, and S. Oh. 2025. "Intent-Based versus GPT-Based Conversational Agents: Benefits and Challenges for Practicing and Assessing Oral Interaction." *TESOL Quarterly* 59:S117–S149. <https://doi.org/10.1002/tesq.v59.S1>.
- Torres, A. B. B., C. Carmichael, W. Wang, M. Paraskevakos, A. Uribe-Quevedo, P. Giles, and J. L. Yawney. 2020. "A 360 Video Editor Framework for Interactive Training." In *8th IEEE International Conference on Serious Games and Applications for Health, SeGAH 2020, Vancouver, BC, Canada, August 12–14, 2020*, 1–7. IEEE.
- Ur, B., E. McManus, M. Pak Yong Ho, and M. L. Littman. 2014. "Practical Trigger-Action Programming in the Smart Home." In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '14*, 803–812. New York, NY, USA: Association for Computing Machinery.
- Vachha, C., Y. Kang, Z. Dive, A. Chidambaram, A. Gupta, E. Jun, and B. Hartmann. 2025. "Dreamcrafter: Immersive Editing of 3D Radiance Fields through Flexible, Generative Inputs and Outputs." In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, 1–13, Yokohama, Japan.
- Wen, C., T. Phung, P. Mehrotra, S. Gulwani, T. Nagashima, and A. K. Singla. 2025. "Exploration vs. Fixation: Scaffolding Divergent and Convergent Thinking for Human-AI Co-creation with Generative Models." ArXiv abs/2512.18388, <https://api.semanticscholar.org/CorpusID:284077702>.
- YiFei, L., and M. K. Othman. 2024. "Investigating the Behavioural Intentions of Museum Visitors towards VR: A Systematic Literature Review." *Computers in Human Behavior* 155:108167. <https://doi.org/10.1016/j.chb.2024.108167>.
- Yigitbas, E., J. Klauke, S. Gottschalk, and G. Engels. 2023. "End-User Development for Interactive Web-Based Virtual Reality Scenes." *Journal of Computer Languages* 74:101187. <https://doi.org/10.1016/j.col.2022.101187>.
- Zarraonandia, T., P. Díaz, A. Montero, and I. Aedo. 2016. "Exploring the Benefits of Immersive End User Development for Virtual Reality." In *Ubiquitous Computing and Ambient Intelligence: 10th International Conference, UCAmI 2016, San Bartolomé De Tirajana*,

Gran Canaria, Spain, November 29–December 2, 2016, *Proceedings, Part I* 10, 450–462. Springer.

- Zhang, L., and S. Oney. 2020. “Flowmatic: An Immersive Authoring Tool for Creating Interactive Scenes in Virtual Reality.” In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*, 342–353, Virtual Event, USA.
- Zhang, L., J. Pan, J. Gettig, S. Oney, and A. Guo. 2024. “VRCopilot: Authoring 3D Layouts with Generative AI Models in VR.” In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology, UIST ’24*. New York, NY, USA: Association for Computing Machinery. <https://doi.org/10.1145/3654777.3676451>.
- Zhao, V., L. Zhang, B. Wang, M. L. Littman, S. Lu, and B. Ur. 2021. “Understanding Trigger-Action Programs through Novel Visualizations of Program Differences.” In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, 1–17, Yokohama, Japan.
- Z. Inc.. 2022. *Zapier*. Online; Accessed February 17, 2022. <https://zapier.com>.

Appendices

Appendix 1. Advances with respect to the State of the Art

Tables A1–A4 summarise the framing of this paper with the literature (Section 2).

Table A1. Summary of Literature Section ‘Rule-based Approaches in EUD’ (Section 2.1).

Reference	Key aspects	Application domain	XR?	Conversational interaction?	Analysis of user utterances/dialogue with agent
Artizzu et al. (2022)	Immersive authoring tool for novice XR developers to find/adapt XR interactions in Unity scenes, by presenting them as natural language (ECA) rules.	Virtual shop, museum	VR	No	n/a
Gallo, Paternò, and Malizia (2024)	Conversational agent assisting users in creating/modifying trigger-action automations via natural language.	IoT / Smart Home	No	Yes (agent)	Limited analysis of errors in chatbot–user interactions
Noura, Heil, and Gaedke (2020)	Grammar-based NLU parser recognising user intents from voice inputs in smart homes.	IoT / Smart Home	No	Goal-oriented voice interaction	Parser evaluated on 253 utterances expressing automation goals, with precision and recall reported.
Liu et al. (2023)	In-Situ Programming (ISP) paradigm for smart home automation.	IoT / Smart Home	No	Natural multimodal voice and gesture interactions elicited via a WoZ study	Qualitative inductive analysis of user-system interactions recorded during an enactment study. A conceptual interaction flow model was derived.
Barricelli et al. (2025)	Conversational creation of automations, with suggestions on rule feasibility and energetic sustainability.	Smart Home, energy-aware	No	Yes (chatbot)	Analysis of suggestion acceptance, chatbot errors and hallucinations.
GreenIFTT (Giudici et al. 2024)	GPT-4-powered app to create/control home automation routines, optimising appliance energy consumption for sustainability.	Home automation, energy optimisation	No	Yes (chatbot)	No
This work	Multimodal conversational agent that helps non-expert users author automation rules using natural language.	Smart Home / Museums automation	Yes	Yes (chatbot)	Systematic analysis of dialogue characteristics in user-agent interaction

A.1. Complete task list

A.1.1. Virtual reality. The task phase required participants to:

- (T1) Explore the environment to get acquainted with its elements;
- (T2) Display an information panel when the user points to an artwork;
- (T3) Turn on an artwork spotlight when getting close to it;
- (T4) Create a complex automation to show additional information on a statue only if the visitor had already seen a painting
- (T5) Create a complex automation to start an audio content when the user is close to an artwork and presses a button;
- (T6) Modify one of the automations they created for a previous task, changing at least one of its parts (trigger, condition or action).

A.1.2. Augmented reality task list. The first task (T1), split into two subtasks (T1.1 and T1.2), aimed to familiarise participants with the experimental setup implemented in the laboratory. The remaining five tasks (T2–T6) presented concrete situations that required users to create appropriate automations:

- (T1) In **T1.1**) describe 2–3 rules you would like to have in your home; in **T1.2**) explore the experimental environment;

Appendix 2. Wizard of Oz study

In this section, we report additional data about the study organisation and results for reproducibility.

- (T2) Define an automation to improve kitchen safety by handling smoke- and gas-related hazards;
- (T3) Ensure that the entrance area becomes illuminated when arriving home in the evening;

Table A2. Summary of Literature Section 'EUD Authoring Tools for XR' (Section 2.2).

Work	XR?	Domain	Interaction paradigm	Orchestration of multiple objects	Authoring platform
IFTTT (Ur et al. 2014)	No	IoT, home automation	Trigger-action GUI	Yes	Web and mobile GUI
EFESTO (Desolda, Ardito, and Matera 2017), TAREME (Ghiani et al. 2017)	No	IoT, home automation	Visual rule editing	Yes	Desktop / web GUI
ImAtHome (Fogli, Peroni, and Stefani 2017)	No	IoT, home automation	Visual rule editing	Yes	Smartphone iOS GUI
VR GREP (Zarraonandia et al. 2016)	VR	General-purpose VR authoring	Multimodal, with limited interaction support	No	VR headset HMD, immersive
Reality Editor (Heun, Hobin, and Maes 2013)	AR	IoT	AR overlay and visual tag-linking	Yes	Handheld mobile device, AR
HoloFlows (Seiger, Gohlke, and Aßmann 2019)	AR	IoT, smart home	Virtual links drawn between physical devices	Yes	AR dedicated device, HoloLens
MagicHand (Sun et al. 2019)	AR	IoT device control	Hand gestures	No, sequential individual control only, no inter-device rules	AR dedicated device, HoloLens
BrickLayer (Stefanidi et al. 2019) / MagiPlay (Stefanidi et al. 2020)	AR	IoT	3D building-block trigger-action authoring with conversational agent as helper	Yes	AR handheld / mobile device
Our work	AR & VR	Smart home, museums	Multimodal conversational interaction	Yes	Smartphone, AR; HMD, VR

Table A3. Summary of Literature Section 'LLM-based Authoring of VR and AR Environments' (Section 2.3).

Work	Authoring target	Purpose of the work	How LLM is used	What is exposed to users
GROMIT (Jennings et al. 2024)	Unity 3D games	Runtime generation of game behaviour in response to player actions	Generates C# Unity behaviour code at runtime	In-scene behaviour changes with short textual descriptions
ImagineAR (J. Lee et al. 2025)	AR	Speech-driven authoring of static outdoor AR scenes, including asset creation and placement	Pipeline of agents acting on scene graphs	Generated or retrieved 3D assets in the scene, with alternatives in AI-assisted mode
Ostaad (Aghel Manesh et al. 2024)	VR	Conversational agent for non-experts to design interactive VR scenes	Translates natural language into executable code for VR scene construction and modification	Updated VR scene, preceded by natural-language previews and code explanations
Dreamcrafter (Vachha et al. 2025)	VR	Scene editing system via photo-realistic scene representations, such as radiance fields	Editing of object style and appearance	Edited scenes with image previews
VRCopilot (Zhang et al. 2024)	VR	Authoring of interior-design 3D layouts exploiting generative AI and multimodal interactions	Processes users' natural-language speech and generates intermediate wireframes	Furnished VR room, preceded by inspectable wireframes, namely 2D views of 3D layouts
LLMR (De La Torre et al. 2024)	MR	Real-time creation and modification of interactive 3D scenes	Pipeline of specialised GPTs generating executable Unity code from natural language	MR scene creation and modifications
LLMER (J. Chen et al. 2025)	XR	Generation of XR interactions and scene changes from users' natural language	Translates natural-language intent into structured JSON executed by dedicated backend modules to create virtual objects and animations in scenes	Interactive XR worlds driven by JSON-defined behaviours
Our work	AR & VR	End-user authoring of persistent ECA automation rules for AR and VR	Conversational LLM agent that elicits user intent, maintains dialogue state, and synthesises ECA rules from natural language	Human-readable natural-language rules that users can inspect, edit, and validate

- (T4) Create a rule to maintain good indoor air quality by controlling an air purifier based on presence and/or time;
- (T5) Define an automation that notifies users when indoor temperatures move outside an optimal range for plant preservation;
- (T6) Modify an existing rule involving a dryer so that it activates only when outdoor conditions do not allow drying outside.

A.2. Structure of the collected rules

A.2.1. Virtual reality. Participants first explored the environment and inspected object states either through the simulated VR information panels or by asking the moderator acting as the chatbot. Two participants (P2 and P6) additionally modified the environment already during exploration by repositioning a wooden sculpture.

Table A4. Summary of Literature Section ‘User recovery strategies from chatbot error’ (Section 2.4).

Work	Agent type	Application domain / scenarios	XR?	Interaction modality	Recovery strategy
Ashktorab et al. (2019)	Intent-based chatbot	Shopping, banking, travel service scenarios	No	Text	Eight system-initiated repair strategies, including repeat, confirmation, options, defer, keyword highlighting, keyword confirmation, and out-of-vocabulary explanation
Følstad and Taylor (2019)	Intent-based chatbot	Financial customer service	No	Text	The chatbot expresses uncertainty and suggests alternative intents when intent misunderstood are likely
Gnewuch and Reinkemeier (2025)	AI/NLP-based chatbot	Insurance customer service	No	Text	Repair messages combining apology, explanation, and guidance when a classifier detects a likely breakdown.
Braggaar et al. (2023)	Intent-based chatbot	Public transportation customer service	No	Text + button selection	Repeat, options, and defer strategies
Moore, An, and Marrese (2024)	Intent-based conversational agent (Watson Assistant)	Small talk, history lesson, coffee ordering scenarios	No	Text + voice	Seven user-repair affordances: definition requests, indexical definition requests, understanding checks, example requests, paraphrase requests, repeat requests, and partial repeat requests.
RELIC (Cheng et al. 2024)	General purpose LLM	Domain-agnostic	No	Text + GUI actions	UI highlights supported, contradicted, or neutral claims and enabling users inspect evidence and edit text.
HILL (Leiser et al. 2024)	General purpose LLM	Domain-agnostic	No	Text + GUI actions	UI highlights problematic passages, provides confidence scores, source URLs, and explanatory details.
FeedbackGPT (Sharma et al. 2026)	General purpose LLM	Domain-agnostic	No	Text + GUI actions	Users can annotate passages, refine vague feedback, and inspect how feedback is interpreted
Gallo et al. / RuleBot++ (Gallo, Paternò, and Malizia 2024)	LLM-based conversational agent	IoT / Smart Home	No	Text	Conversational automation refinement, automation verification, natural-language automation summary, and confirmation before saving
Our work	LLM-based conversational agent	Smart Home, Museums	Yes	Voice + pointing + gaze	Conversational automation refinement, automation verification, natural-language automation summary, and confirmation before saving

Across the study, participants created 112 rules, including 98 newly defined automations and 14 modifications. Among these, 100 rules satisfied the structural validity requirements. Twelve of the fourteen participants completed all tasks; P3 did not define rules for T4 and T5, while P5 did not complete T3.

For the simple rule-definition tasks (T2 and T3), all participants except P5 created the requested rules containing at least one event and one action. In total, participants formulated 37 rules for these tasks, including 10 rules with multiple events or actions. Three participants (P8, P9, and P13) created additional rules beyond the requested tasks.

For the complex rule-definition tasks (T4 and T5), participants produced 50 automations containing triggers, conditions, and actions. Three participants (P1, P4, and P8) constructed complex automations by combining multiple simpler rules. Participants frequently omitted explicit references to trigger subjects, particularly in complex rules.

Task T6 required participants to modify an existing rule, resulting in 14 rule modifications. Two participants (P2 and P6) chose to create a new rule before making modifications. Most participants changed the whole rule instead of just the part that needed modification. Only in three sessions did they adjust just one component, mainly the action.

Table A5. Rule groups with their structures and frequencies in the WoZ and chatbot prototype studies across AR and VR settings (E = event, C = condition, A = action).

Group #	Structure %	WoZ VR		WoZ AR		Prototype VR		Prototype AR	
		#	%	#	%	#	%	#	%
Event-first	E-based (E ⁺)	0	0.0%	0	0.0%	0	0.0%	1	2.1%
	EA-based (E ⁺ A ⁺)	62	55.3%	57	35.1%	34	39.5%	11	23.4%
	EAC-based (E ⁺ C ⁺ A ⁺)	6	5.3%	10	6.1%	7	8.1%	0	0.0%
	ECA-based (E ⁺ C ⁺ A ⁺)	4	3.6%	17	10.4%	6	6.9%	1	2.1%
Action-first	A-based (A ⁺)	10	8.9%	0	0.0%	3	3.4%	0	0.0%
	AE-based (A ⁺ E ⁺ A ⁺)	18	16.0%	46	28.3%	16	18.6%	17	36.1%
	AEC-based (A ⁺ E ⁺ C ⁺)	2	1.8%	12	7.4%	3	3.4%	6	12.7%
	ACE-based (A ⁺ C ⁺ E ⁺)	1	0.9%	7	4.3%	3	3.4%	1	2.1%
Condition-first	C-based (C ⁺)	0	0.0%	0	0.0%	7	8.1%	7	14.8%
	CAE-based (C ⁺ A ⁺ E ⁺)	0	0.0%	3	1.8%	3	3.4%	1	2.1%
	CA-based (C ⁺ A ⁺)	2	1.8%	0	0.0%	2	2.3%	0	0.0%
	CEA-based (C ⁺ E ⁺ A ⁺)	7	6.2%	10	6.1%	2	2.3%	2	4.2%

Notes: The symbols ⁺ and * indicate repeated occurrences (1 or more, and 0 or more, respectively).

Table A6. Task-level performance and error patterns in the AR and VR settings with Tell-XR.

Errors																
Task	Time		Rounds		System						User					
	$\bar{x}$ (mm.ss)	σ (s)	$\bar{x}$	σ	Total	RE	WC	HC	#	%	UD	WU	IU	#	%	
VR	1	5 m 21	154	6	2	22	11	3	2	16	72,7	3	3	0	6	27,3
	2	3 m 44	89	4	2	22	12	4	1	17	77,3	1	4	0	5	22,7
	3	7 m 25	146	9	3	29	15	9	2	26	89,7	3	0	0	3	10,3
	4	6 m 44	216	7	4	42	18	11	5	34	81	1	7	0	8	19
	5	4 m 42	189	6	2	18	4	2	2	8	44,4	1	9	0	10	55,6
	6	4 m 40	167	5	2	10	4	4	1	9	90	0	1	0	1	10
Errors																
Task	Time		Rounds		System						User					
	$\bar{x}$ (mm.ss)	σ (s)	$\bar{x}$	σ	Total	RE	WC	HC	#	%	UD	WU	IU	#	%	
AR	1	4 m 13	148	10	6	38	5	12	2	19	50	5	3	11	19	50
	2	3 m 47	156	9	4	23	3	7	2	12	52,2	1	1	9	11	47,8
	3	3 m 34	160	10	5	27	6	6	2	14	51,9	1	0	12	13	48,1
	4	2 m 21	64	7	3	26	3	6	0	9	34,6	1	10	6	17	65,4
	5	2 m 46	60	8	3	28	0	3	3	6	21,4	0	9	13	22	78,6
	6	3 m 21	104	10	3	38	7	10	1	18	47,4	1	4	15	20	52,6

Notes: Each condition comprised the same six tasks. For each task, the table reports the mean completion time (in minutes and seconds) and standard deviation (in seconds), the mean number of conversational rounds (i.e. user–chatbot turns) and its standard deviation, and the distribution of errors derived from our thematic analysis of the interaction logs. Error counts include the total number of errors identified for the task, followed by a breakdown into system-generated errors and user-generated errors. System errors include RE (speech recognition errors), WC (wrong chatbot response errors), and HC (hallucinations). User errors include UD (user digressions), WU (wrong user responses), and IU (inconsistent user responses). For both system and user categories, the table reports the absolute frequency (#) and the proportion of errors (%) relative to the total errors for that task.

The analysis of rule structures shows that participants preferred to formulate rules starting with events (see Table A5). Simple event-action (EA) sequences appeared 62 times. We observed 16 rules that included an event followed by multiple actions (EAA, EAAA, EAAAA). Event-action-condition (EAC) structures occurred less often, while event-condition-action (ECA) structures were rare, indicating that participants tended to place conditions later in the rule. Participants also frequently used action-first formulations, with simple action-event (AE) rules occurring 18 times. Further complex structures, such as sequences involving multiple actions and/or conditions, appeared less frequently, and condition-first structures occurred least often. These results indicate that participants conceptualise automation primarily as direct cause-effect relations, favouring event-driven interactions. This tendency is further illustrated by three participants who divided automation requiring conditions into multiple trigger-action rules.

The analysis of sentence structures shows that participants most often defined events using direct statements without specific keywords (36 occurrences) or implicit formulations (20 occurrences). They occasionally used the terms ‘if’ (14 occurrences) and ‘when’ (12 occurrences), suggesting flexibility in describing events. Participants expressed conditions mainly with ‘if’ (9 occurrences), complemented by temporal expressions (‘in the moment that’ -- 2) and negations (‘must not’ -- 1), reflecting diverse ways to encode constraints. Participants predominantly used action verbs related to media control (‘play’ -- 24, ‘show’ -- 16) and interactive commands such as ‘teleport’ (7) and ‘rotate’ (5), indicating engagement with dynamic virtual elements. Verb forms varied, with impersonal forms appearing 60 times and first-person forms 55 times, while conditional expressions occurred 41 times, showing that participants preferred flexible rule definitions. They also included spatial references (38 occurrences) and comparative

expressions (20 occurrences), emphasising context-awareness in their actions.

Participants occasionally employed modalities other than speech to express rules. They used pointing most frequently (42 times), typically to indicate objects or locations relevant to automation. Pantomimes occurred 24 times, with participants acting out expected automation effects, such as pressing a button or turning on a light. Body movements were less common (5 occurrences), where participants performed gestures to simulate interactions (e.g. stepping forward to trigger an event). These findings show that participants combined verbal and non-verbal communication to define automation rules.

A.2.2. Augmented reality. Overall, participants created 162 final, non-ambiguous automations during the AR formative test, 78 of which are complex rules.

Participants defined complex rules 78 times. Among these, the most frequent structures placed the event at the beginning of the rule: $E^+C^+A^*$ (17), $EA^+C^+A^*$ (10), CE^+A^+ (10), and A^+CE^+ (7). Table A5 reports all observed rule structures, grouped according to the element appearing first (event, condition, or action). The distribution highlights a strong preference for event-first formulations (84), followed by action-first ones (65). Percentages refer to the total number of final rules collected (162).

All participants defined at least one automation for every task, and in several cases they produced more than one automation within the same task. In particular, for Tasks T4 and T5 (which explicitly required the creation of a single complex rule), users sometimes opted to define two separate rules, or even three, as observed for three participants in T4. Consequently, across T4 and T5 the 15 users produced a total of 47 rules (25 simple and 22 complex), rather than the 30 complex rules originally expected. This outcome indicates a tendency to reduce the perceived complexity of a single

comprehensive automation by decomposing it into multiple simpler rules, each addressing a specific facet of the task, instead of combining all elements within a single complex rule.

Task T6 required users to extend an existing simple rule by adding a condition. In this case, we expected users to mention only the new rule element. However, participants typically reformulated the full modified rule, sometimes in a partial manner. For example, some users restated only the action component of the original rule, implicitly referring to the triggering event without explicitly repeating it.

We further examined the linguistic expressions adopted by participants to describe rule events, conditions, and actions. Event descriptions most frequently relied on the term ‘when’ and its minor variants (e.g. ‘whenever’, ‘when not’), accounting for 34.32% of the 169 event occurrences. Participants also introduced events using ‘if/if not’ (16.56%), or by specifying a time reference, such as ‘from’ <time> or ‘at’ <time> (10.05%). Less frequently, they used expressions like ‘in case of’ or ‘only in case’ (8.87%).

When expressing conditions, users predominantly employed ‘if/if not’ or ‘only if’, which accounted for 47.06% of the 68 condition occurrences. Additional formulations included generic expressions such as ‘in case’, as well as season-related constraints (e.g. ‘only during the summer’, 7.35%) and temporal intervals (e.g. ‘from x to y’ or ‘in the time interval from x to y’, 5.88%). Action descriptions most often appeared in the second person, with users directly requesting the agent to perform an operation (29.28% of 181 action occurrences). Alternatively, participants used first-person expressions such as ‘I’d like to’ or ‘I want to’ (23.20%). Third-person formulations (e.g. ‘It must’, ‘It will’) occurred less frequently (11.60%).

Appendix 3. Chatbot prototype

A.3. Complete task list

A.3.1. Virtual reality. In VR, the task phase required participants to:

- (T1) Automate spotlights for a painting when visitors approach (*Simple*);
- (T2) Display an info panel when a button is pressed (*Simple*);
- (T3) Create an automation of their choice (Free);
- (T4) Unlock hidden information with a related historical object (*Complex*);
- (T5) Play multimedia content near Beethoven’s statue if playback is enabled (*Complex*);
- (T6) Modify the simple rule from T2 to create a complex automation (*Modify*).

A.3.2. Augmented reality. In AR, the task phase required participants to:

- (T1) Create an automation to prevent kitchen hazards, such as forgetting a pot on the stove (*Simple*);
- (T2) Automate the front door lighting to ensure it turns on when the user arrives home in the dark (*Simple*);
- (T3) Create an automation of their choice (Free);

- (T4) Automate the air purifier to remain on while the user is in the living room (*Complex*);
- (T5) Create an automation to regulate temperature for houseplants by opening a window when it is too hot on weekends (*Complex*);
- (T6) Modify the simple rule from T2 to create a complex automation (*Modify*).

A.4. Structure of the collected rules

A.4.1. Virtual reality. We analysed 86 rules collected in the VR setting (see Table A5). Most users favoured event-first formulations: 26 rules followed a simple event–action (EA) pattern. Extended event–action structures appeared less frequently (EAA: 2 cases; EAAE: 1 case). Conditions (C) occurred rarely within event-based structures, with EAC appearing four times and ECA five times, indicating that users typically introduced conditions later in the rule definition process. Action-first formulations appeared more often than expected, with 15 AE rules, 3 standalone action (A) rules, and 7 condition-based rules. This pattern suggests that some users conceptualised automations primarily in terms of actions or conditions rather than event-driven triggers. These findings align with prior work highlighting a preference for simple cause–effect relationships and an avoidance of complex logic unless required. Unlike the previous study, where chatbot assistance led users to simplify complex rules, the present results show a wider range of rule structures, suggesting that LLM-based responses supported more diverse problem-solving strategies.

We further examined the terminology participants used in VR to express events, conditions, and actions, considering only their own utterances. We identified 79 event-related expressions, 35 condition-related expressions, and 97 action-related expressions. Participants most frequently used the term ‘when’ (52 occurrences) to introduce events, reflecting a temporal interpretation of triggers. Less frequent alternatives included direct statements (6), ‘every’/‘once’ (5), ‘if’ (5), and conjunctions such as ‘and’ (4). For conditions, the most common terms were ‘when’ (8), ‘and’ (7), and ‘if’ (4), followed by direct statements (3) and specific constraints. For actions, participants primarily relied on verbs such as ‘play’ (17), ‘turn on’ (17), and ‘activate’ (13), followed by ‘show’ (9) and ‘start’/‘appear’ (5 each). These choices indicate a focus on clear cause–effect formulations, with temporal expressions for events and relatively simple expressions for conditions.

Regarding verb structure, participants strongly preferred the second person (83 occurrences) and present tense (112), formulating rules mainly as commands directed to the chatbot. The frequent use of imperative forms (69) further supports this tendency. Modal verbs such as ‘can’ (11), ‘do’ (10), ‘must’ (6), and ‘want’ (6) appeared less often and introduced some variability in phrasing and perceived execution flexibility.

A.4.2. Augmented reality. Participants created a total of 70 rules (32 simple, 38 complex). In 20 cases (10 simple, 10 complex), users accepted the chatbot’s proposed rule without changes. This occurred most frequently in T3 (8 cases), followed by T1 (5), T2 (3), T4 (2), and T5 and T6 (1 each). The open-ended nature of T3 likely contributed to this pattern. In T6, three users modified only an event parameter of

an existing rule. Eight tasks did not result in a saved rule due to inconsistent user responses or chatbot errors.

We analysed the structure of all collected rules (Table A5), distinguishing between user-defined rules and chatbot-proposed rules accepted by users. Action-first structures dominated (24 rules), indicating a focus on intended outcomes; the most common pattern was $A^+E^+A^*$ (17 rules). Event-first structures appeared less frequently (13 rules), mostly following the E^+A^+ pattern (11 rules). This contrasts with earlier formative findings that highlighted event-driven formulations, suggesting a shift when users interact with a fully functional AR prototype. Condition-first rules occurred in 10 cases, including 7 rules consisting solely of conditions (C^+), all related to T6. In these cases, users stated only the new condition, leaving events and actions implicit.

We analysed users' terminology for events, conditions, and actions, considering only their own utterances. We identified 38 event expressions, 17 condition expressions, and 38 action expressions. Consistent with the formative study, users relied primarily on 'when' (24) and 'if' (5) for events, and on expressions such as 'only when', 'only', 'when', and 'during [the weekend]' for conditions.

We also grouped action verbs by semantic meaning. The verb 'to turn' occurred most frequently (15), followed by 'to

open' (12), reflecting the devices involved in T5. Verbs such as 'to notify' and 'to set' appeared four times each, while 'to be' and 'to close' appeared once. Users often adopted structures combining want/wish with the subjunctive and an explicit device reference (11 cases), e.g. "I would like the purifier to turn on". In nine cases, users directly addressed the chatbot (e.g. "I would like you to turn on the purifier"). Other formulations referred to automations more abstractly or relied on infinitive verb forms.

A.5. Error count

Appendix 4. Example dialogues and their resulting rules with Tell-XR

In this section, we provide a few examples of user chatbot dialogues, one for each task, together with the corresponding automation. The original dialogues were conducted in Italian and have been adapted into English for this article.

A.6. Virtual museum

A.7. Smart home

	[VR10]: Until next episode.
	[Tell-XR]: If you need help in the future, don't hesitate to come back. See you soon!
	[VR10]: Ei bot, set up lighting when people approach the painting, The Great Wave.
	[Tell-XR]: Do you want to create an automation that turns on the "The Great Wave" spotlight when the player approaches the "The Great Wave" painting?
	[VR10]: Yes.
	[Tell-XR]: Do you want to add any condition?
	[VR10]: No.
	[Tell-XR]: The automation you defined is "Turn on the 'The Great Wave' spotlight when the player approaches the 'The Great Wave' painting". Do you want to save and export the automation?
	[VR10]: Yes, that's fine.
	[Tell-XR]:
<pre>{ "trigger": { "subject": "player", "verb": "interacts with", "obj": "painting_thegreatwave" }, "conditions": null, "actions": [{ "subject": "spotlight_thegreatwave", "verb": "turns", "obj": "on" }], "description": "Turn on the 'The Great Wave' spotlight when the player approaches the 'The Great Wave' painting" }</pre>	
	[Tell-XR]: Automation saved. It turns on the "The Great Wave" spotlight when the player approaches the painting. What's next: create a new automation or modify an existing one?

Figure A1. VR Task 1 dialogue example. The user intended to create an automation that turns on the spotlight for The Great Wave when a visitor approaches the painting. We assigned the created automation a score of 6 because the user successfully completed the task, and the chatbot selected the correct services for both the trigger and the action specified by the user. It is worth noting that, although an STT error occurred at the beginning of the dialogue, it did not affect the next rounds.

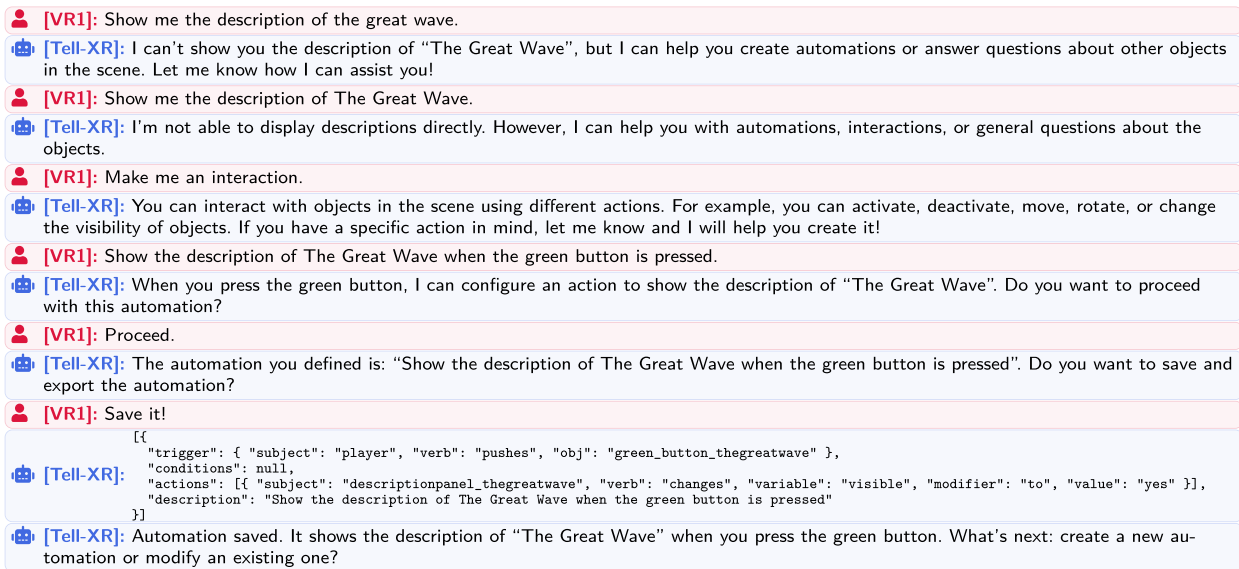


Figure A2. VR Task 2 dialogue example. After an initial exchange in which the user appeared to give direct commands to the chatbot rather than formulate an automation intent, the chatbot clarified what it could support. The user then specified the intention to show the description panel of the artwork when the green button is pressed. In this case, we assigned a score of 6 to the generated automation: the user successfully completed the task, and the chatbot selected the correct services for both the trigger and the action specified by the user.

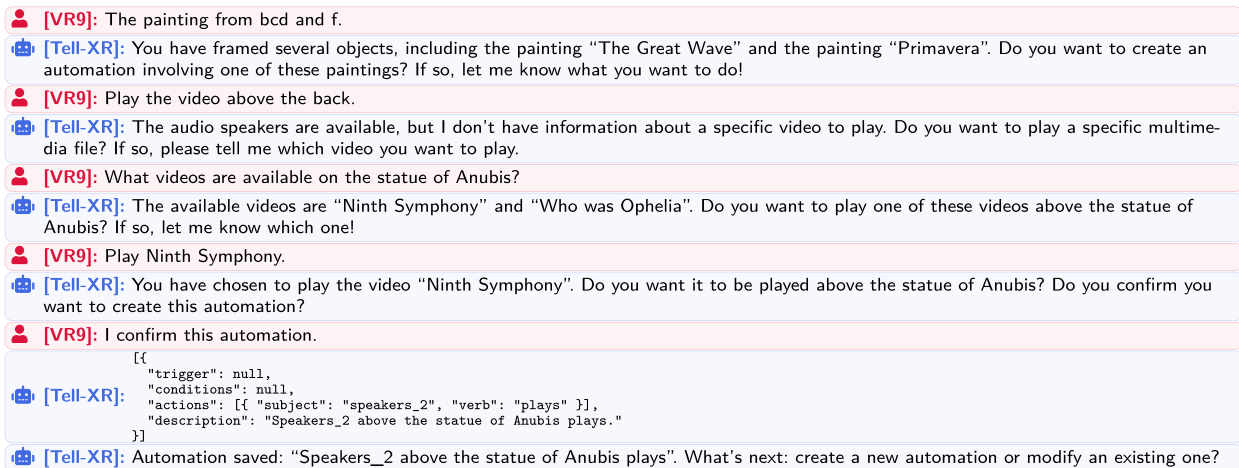


Figure A3. VR Task 3 dialogue example. This dialogue was initially disrupted by several STT errors, after which the user specified the intention to play the video Ninth Symphony above the statue of Anubis, without indicating the event that should trigger these actions. The chatbot asked for confirmation without requesting this missing information, and the generated automation was incorrect, as it lacked a trigger and failed to set Ninth Symphony as the audio source for Speakers_2. We assigned a score of 1 to the resulting automation.

[VR4]: When a visitor approaches the bust of Nefertiti and puts the sceptre in the container, the information about the artwork is shown.

[Tell-XR]: You mentioned wanting to create an automation that shows information about the artwork when a visitor approaches the bust of Nefertiti and puts the sceptre in the container. To proceed, we could configure the approach-to-the-bust-of-Nefertiti event and the action of showing the information on the Nefertiti screen. Do you want to add conditions, for example that the visitor is actually near the bust or that the sceptre is in the container? How do you want to proceed?

[VR4]: The necessary condition is that the sector is put in the container.

[Tell-XR]: So, the automation will be configured to show information about the artwork when a visitor approaches the bust of Nefertiti and puts the sceptre in the container. The event will be the approach to the bust of Nefertiti, the action will be showing the information on the Nefertiti screen, and the condition will be that the sceptre is in the container. Do you want to add other conditions or events, or can we proceed with this configuration?

[VR4]: Proceed with this configuration.

[Tell-XR]: The automation you defined is: show information about the artwork when a visitor approaches the bust of Nefertiti and puts the sceptre in the container. Do you want to save and export the automation?

[VR4]: Yes.

[Tell-XR]:

```
{
  "trigger": { "subject": "player", "verb": "interacts with", "obj": "nefertiti_statue" },
  "conditions": { "component": "socket_egypt_sceptre_nefertiti", "property": "contains", "symbol": "==", "compareWith": "egypt_sceptre" },
  "actions": [{ "subject": "descriptionpanel_ophelia", "verb": "shows" }],
  "description": "Show information about the artwork when a visitor approaches the bust of Nefertiti and the sceptre is placed into the container."
}
```

[Tell-XR]: The automation has been saved successfully. It shows information about Ophelia when the visitor approaches the bust of Nefertiti and puts the sceptre in the container. What's next? Do you want to create a new automation or modify an existing one?

Figure A4. VR Task 4 dialogue example. The user intended to show information about the bust of Nefertiti when a visitor approaches it, provided that the sceptre has been placed in the container. In this example, the generated rule was assigned a score of 4 because, at the confirmation stage, the chatbot strictly adhered to the user's use of the term 'artwork', interpreting it as referring to one of the paintings in the environment rather than to the statue itself. This error occurs despite the fact that, in the previous step, the chatbot had correctly summarised the user's intent.

[VR7]: In front of the bust of Beethoven, you can play the audio file of the Ninth Symphony and the video of the same Ninth Symphony. No, I'm still listening to myself, so...

[Tell-XR]: You mentioned wanting to play the audio file of the Ninth Symphony and the video of the Ninth Symphony when in front of the bust of Beethoven. I can help you configure this automation. Do you want to add specific conditions? How do you want to proceed?

[VR7]: When the user is in front of the statue of Beethoven you can start both the audio file and the video file.

[Tell-XR]: So you want the audio and video of the Ninth Symphony to start when the user is in front of the statue of Beethoven. You have already specified the event and the actions. Do you want to add other conditions? How do you want to proceed?

[VR7]: What other conditions could I add?

[Tell-XR]: For example, you can check whether the statue of Beethoven is currently active, or require the visitor to be in a specific position.

[VR7]: No. Proceed with this configuration.

[Tell-XR]: The automation you defined is: when the user is in front of the statue of Beethoven, start both the audio file and the video file of the Ninth Symphony. Do you want to save and export the automation?

[VR7]: Yes, save the automation.

[Tell-XR]:

```
{
  "trigger": { "subject": "player", "verb": "interacts with", "obj": "beethoven_statue" },
  "conditions": null,
  "actions": [{ "subject": "beethoven_speakers", "verb": "plays" }, { "subject": "beethoven_monitor", "verb": "plays" }],
  "description": "When the user is in front of the statue of Beethoven, start both the audio file and the video file of the Ninth Symphony"
}
```

[Tell-XR]: The automation has been saved. It will start the audio and video of the Ninth Symphony when the user is in front of the statue of Beethoven. What's next? Do you want to create a new automation or modify an existing one?

Figure A5. VR Task 5 dialogue example. In this dialogue, the user asked to start both the audio and video of the Ninth Symphony when the visitor is in front of the statue of Beethoven, without specifying the required condition for enabling media playback. The chatbot created a rule that correctly addressed the user's stated requirements, but the user omitted a necessary condition. For this reason, we assigned the created rule a score of 5.

 [VR8]: Show the description panel of The Great Wave only when the light is on.

 [Tell-XR]: You want to modify the automation so that the description panel of The Great Wave is shown only when the lamp is on. Would you like to add any other conditions or events to this automation?

 [VR8]: No.

 [Tell-XR]: So, the automation will be modified to show the description panel of The Great Wave only when the lamp is on. Would you like to save this automation?

 [VR8]: Save.

 [Tell-XR]: Do you confirm that you want to save the modified automation that shows the description panel of The Great Wave only when the lamp is on?

 [VR8]: Yes.

 [Tell-XR]:

```

    [{"trigger": { "subject": "player", "verb": "pushes", "obj": "green_button.thegreatwave" },
      "conditions": [{ "component": "spotlight.thegreatwave", "property": "turn", "symbol": "==", "value": "yes" }],
      "actions": [{ "subject": "spotlight.thegreatwave", "property": "on", "symbol": "==", "compareWith": "yes" }],
      "description": "Show description of The Great Wave",
      "id": "3474fe16-c50d-49c5-8dc5-284a8b51c0b3"
    }]
  
```

 [Tell-XR]: Automation modified: the description panel of The Great Wave will be shown when the lamp is on. What would you like to do next: create a new automation or modify an existing one?

Figure A6. VR Task 6 dialogue example. The participant asked to update the existing automation created in the first task so that the description panel of The Great Wave is shown only when the light is on. In this case, we assigned a score of 6 to the generated automation: the participant completed the task successfully, and the chatbot correctly identified the request as an update, added the condition correctly, and preserved the original trigger and action.

 [AR9]: Hey Bot

 [Tell-XR]: Hello! How can I help you with your smart home today? Would you like to create an automation?

 [AR9]: On my smartphone, could you notify me when the stove is on and my presence has not been detected in the kitchen for more than 10 minutes?

 [Tell-XR]: What message would you like to receive on your smartphone? Would you like to add any other conditions or actions?

 [AR9]: A message saying, "Warning, you left the stove on."

 [Tell-XR]: Would you like to save this automation with the message "Warning, you left the stove on" and the user's presence has not been detected in the kitchen for more than 10 minutes?

 [AR9]: Yes

 [Tell-XR]:

```

    [{"trigger": {
      "platform": "state",
      "entity_id": "binary_sensor.shellygas_3c6105f7467b_gas",
      "to": "on",
      "for": "00:10:00"
    },
      "conditions": [{
        "condition": "state",
        "entity_id": "binary_sensor.zy_m100_wificun_zai_gan_ying_qi_occupazione",
        "state": "off"
      }],
      "actions": [{
        "service": "notify.mobile_app",
        "data": { "message": "Warning, you left the stove on" }
      }],
      "description": "Warning stove on",
      "id": "3474fe16-c50d-49c5-8dc5-284a8b51c0b3"
    }]
  
```

 [Tell-XR]: The automation has been successfully created.

Figure A7. AR Task 1 dialogue example. The user requested the creation of an automation to receive a notification on their smartphone when the stove is on and the user's presence has not been detected in the kitchen for more than 10 minutes. The notification message must be 'Warning, you left the stove on'. The automation has been successfully created. We assigned a score of 4 to the generated automation because the user was correct but the chatbot was partially correct.

[AR8]: Hey Bot.

[Tell-XR]: Hi! How can I help you today with your smart home? Do you want to create an automation?

[AR8]: I would like to create an automation for the lights.

[Tell-XR]: You can turn on the bulb when you enter the house or when motion is detected. What would you prefer to do?

[AR8]: I would like it to turn on both in motion and when I arrive.

[Tell-XR]: You can set the automation so that the bulb turns on when motion is detected or when your smartphone reports that you have arrived home. Do you want to save this automation or add something else?

[AR8]: I want to add any movement.

[Tell-XR]: Do you want the bulb to turn on with any movement in the house or only in specific rooms?

[AR8]: With any movement in the house.

[Tell-XR]: Perfect! I will set the automation to turn on the bulb with any movement detected in the house. Do you want to proceed?

[AR8]: Yes, proceed.

[Tell-XR]: The defined automation turns on the bulb when any movement is detected in the house or when the user's smartphone reports that they have arrived home.

[AR8]: Yes, save the automation.

```

{
  "description": "Turn on bulb with motion or arrival at home",
  "trigger": [{ "platform": "state", "entity_id": "binary_sensor.motion", "to": "on" }, { "platform": "state", "entity_id": "binary_sensor.zy_m100_wificun_zai_gan_ying_qi_occupazione", "to": "on" }, { "platform": "state", "entity_id": "sensor.gps_redminote8", "to": "home" } ],
  "action": [{ "service": "light.turn_on", "entity_id": "light.shellycolorbulb_3494546b686e" } ],
  "id": "6x6vax3c"
}

```

[Tell-XR]: The automation has been created successfully: the bulb will turn on with any movement detected in the house or when you arrive home.

Figure A8. AR Task 2 dialogue example. The user asked to create an automation to turn on the bulb when motion is detected or when they arrive home. They specified that the bulb should turn on with any movement in the house. The automation was configured to turn on the bulb both when motion is detected through motion sensors and when the user's smartphone reports that they have arrived home. The user confirmed that they wanted to proceed with this automation and asked to save it. In this case, we assigned a score of 6 to the generated automation.

[AR5]: Hey Bot.

[Tell-XR]: Hi! How can I help you today with your smart home? Do you want to create an automation?

[AR5]: Yes.

[Tell-XR]: What automation goal do you have in mind? Security, comfort, energy saving, or something else?

[AR5]: Energy saving.

[Tell-XR]: You can automate the bulb switching off when there is no movement in the living room. Do you want to proceed?

[AR5]: Yes, thank you.

[Tell-XR]: Perfect! I will set the automation to turn off the bulb when the motion sensor in the living room does not detect movement. Do you want to add a specific inactivity time before switching it off?

[AR5]: Yes, thank you.

[Tell-XR]: How much inactivity time do you want to set before turning off the bulb?

[AR5]: Seconds.

[Tell-XR]: How many seconds of inactivity do you want to set before turning off the bulb?

[AR5]: Forty.

[Tell-XR]: I will set the bulb to switch off after 40 seconds of inactivity. Do you want to add other conditions?

[AR5]: No, thank you.

[Tell-XR]: The defined automation turns off the bulb at the entrance when the motion sensor in the living room does not detect movement for 40 seconds. There are no time limitations or additional conditions.

```

{
  "description": "Bulb switch-off for energy saving",
  "trigger": [{ "platform": "state", "entity_id": "binary_sensor.living_room_motion", "to": "off", "for": "00:00:40" } ],
  "action": [{ "service": "light.turn_off", "entity_id": "light.shellycolorbulb_3494546b686e" } ],
  "mode": "single",
  "id": "wnbWB3iT"
}

```

[Tell-XR]: The automation has been created successfully to turn off the bulb after 40 seconds of inactivity in the living room.

Figure A9. AR Task 3 dialogue example. During the conversation, the user expressed the desire to create an automation for energy saving. The automation turns off the entrance bulb when the motion sensor in the living room does not detect movement for 40 seconds. The user did not want to add time limitations or additional conditions. The automation was configured successfully. In this case, we assigned a score of 6 to the generated automation.

[AR7]: Hey bot.

[Tell-XR]: Hi! How can I help you create an automation for your smart home?

[AR7]: I would like to create an automation with the air purifier.

[Tell-XR]: You can turn on the air purifier when the air quality is poor. Do you want to proceed with this automation?

[AR7]: I want to create another one.

[Tell-XR]: You can turn off the air purifier when there is nobody at home. Do you want to proceed with this automation?

[AR7]: I want to create another one.

[Tell-XR]: You can turn on the air purifier when the PM2.5 level exceeds a certain threshold. Do you want to proceed with this automation?

[AR7]: The air purifier should turn off when the person leaves. When there is nobody.

[Tell-XR]: You can turn off the air purifier when there is nobody at home. Do you want to proceed with this automation?

[AR7]: Yes.

[Tell-XR]: Sorry for the repetition. Do you want to proceed with the automation to turn off the purifier when there is nobody?

[AR7]: Exactly.

[Tell-XR]: Do you want to add a notification when the purifier turns off? If yes, what content?

[AR7]: No, no notification.

[Tell-XR]: The automation turns off the air purifier when there is nobody at home.

[AR7]: Save.

```

{
  "description": "Turn off purifier when home is empty",
  "trigger": [{ "platform": "state", "entity_id": "binary_sensor.zy_m100_wificun_zai_gan_ying_qi_presence", "to": "false" }],
  "condition": [],
  "action": [{ "service": "fan.turn_off", "entity_id": "fan.xiaomi_cpa4_d0bb_switch_status" }]
}

```

[Tell-XR]: The automation to turn off the air purifier when there is nobody at home has been created successfully.

Figure A10. AR Task 4 dialogue example. The user expressed the desire to create an automation for the air purifier. After some proposals, the user decided to turn off the purifier when there is nobody at home, without receiving notifications. The automation was created successfully. We assigned the score 5 because the chatbot correctly implemented the user's request, but the user-defined automation is only partially correct for the task.

[AR9]: Hey Bot.

[Tell-XR]: Hello! How can I help you with automations for your smart home?

[AR9]: The indoor temperature exceeds 25°C.

[Tell-XR]: Would you like to set up an automation for when the indoor temperature exceeds 25°C? If so, what action would you like to be performed?

[AR9]: Yes, I'd like you to open the living room window.

[Tell-XR]: I can't open the window automatically. Would you like to receive a notification to open it manually?

[AR9]: Yes, thanks, that's fine.

[Tell-XR]: The defined automation specifies that when the indoor temperature exceeds 25°C, the living room window will be opened. Would you like to save it?

[AR9]: Yes, thanks, save it.

```

{
  "description": "Notify if temperature exceeds 25°C",
  "trigger": [{ "platform": "numeric_state", "entity_id": "sensor.netatmo_air_quality_soggiorno_temperatura", "above": 25 }],
  "condition": [],
  "action": [{ "service": "notify.notify", "data": { "message": "The temperature has exceeded 25°C. I suggest opening the living room window." } }],
}

```

[Tell-XR]: I have created an automation to notify you when the indoor temperature exceeds 25°C, suggesting opening the living room window.

Figure A11. Task 5: During the conversation, the user expressed the desire to set up an automation for your smart home. The automation defines that when the indoor temperature exceeds 25°C, suggesting to open the living room window. After discussing the possibilities, the bot proposes a notification to manually remind the user to open the window. We assigned this automation a score of 1 because both the user and the bot made mistakes. The user failed to specify that the automation should be executed only on weekends, as required by the task. The bot, on the other hand, hallucinated by stating that the window could not be opened automatically, even though this is actually possible. Instead, it suggested sending a notification.

[AR5]: Hey Bot.

[Tell-XR]: Hello! How can I help you with your smart home today? Would you like to create an automation?

[AR5]: No, I want to modify an automation.

[Tell-XR]: Which automation would you like to modify?

[AR5]: The entrance light one.

[Tell-XR]: I found the automation "Turn on the entrance light when near home". Would you like to add something to it?

[AR5]: I want to add a condition.

[Tell-XR]: Would you like to add a condition so that the light turns on only at certain times or on certain days?

[AR5]: I want the automation to be active only from 5:00 PM onward.

[Tell-XR]: Would you like the automation to be active only from 5:00 PM onward every day?

[AR5]: Yes, thank you.

[Tell-XR]: Perfect, I'll add the condition to activate the automation only from 5:00 PM onward. Would you like to save the changes?

[AR5]: Yes.

[Tell-XR]: The automation turns on the entrance light when the user is within 1 km of home, but only if the time is after 5:00 PM.

```

{
  "id": "GtKAJKLF",
  "description": "Turn on entrance light on return",
  "trigger": [{ "platform": "state", "entity_id": "sensor.gps_redminote8", "below": "1" "from": "home" }],
  "condition": [{ "condition": "time", "after": "17:00" }],
  "action": [{ "target": { "entity_id": "light.shellycolorbulb_3494546b686e" }, "action": "light.turn_on" }],
  "mode": "single"
}

```

[Tell-XR]: I have updated the automation to turn on the entrance light only after 5:00 PM when you are near home.

Figure A12. Task 6: The user requested to modify a lighting-related automation. After identifying the automation 'Turn on the entrance light when near home', the user asked to add a condition so that the automation is activated only after 5:00 PM. The automation was updated accordingly, and the user confirmed that they wanted to save the changes. In this case, we assigned a score of 6 to the generated automation.