EASE: EFFICIENT HUMAN–AGENT COMMUNICATION UNDER EVOLVING USER PREFERENCES

Anonymous authors
Paper under double-blind review

ABSTRACT

Agentic AI systems can execute increasingly complex real-world tasks, yet their transition from passive responders to active task coordinators raises a central communication challenge: how to efficiently incorporate necessary human input without overburdening the user. Existing systems often fail to adapt interaction forms and progress-reporting strategies as user preferences evolve, leading to redundant elicitation, over-automation, and verbose operational traces. In this paper, we propose EASE, an adaptive communication framework that enables granular control over user engagement, interaction form, and progress exposure. By mapping user preference stability, intervention signals, and task risk to a fine-grained automation spectrum, EASE elicits necessary user input with reduced user effort. To support timely intervention without cognitive overload, EASE surfaces only decision-relevant execution states while suppressing low-level operational details. Extensive experiments show that EASE consistently reduces time cost and cognitive load compared with strong baselines across no-history, stable-preference, and preference-drift settings. These results demonstrate that efficient human–agent communication is achieved not by maximizing autonomy alone, but by dynamically calibrating when, how, and to what extent users should be engaged.

1 INTRODUCTION

By integrating external tools like search engines and calendars, artificial intelligence (AI) models are transitioning from passive responders to active agents capable of driving tangible outcomes (Wang et al., 2026; Yin et al., 2024; Abou Ali et al., 2025). Magentic-UI, a recent agentic system developed by Microsoft Research, exemplifies this shift by supporting real-world tasks such as booking flights and scheduling meetings (Mozannar et al., 2025). Despite their promise of autonomous execution, existing agentic systems typically maintain static architectures and fixed functionalities throughout sessions (Fang et al., 2025; Gao et al., 2024; Yang et al., 2025). Without the ability to evolve with sustained human–agent interactions, these systems often require the user to repeatedly specify their preferences, leading to longer task completion times and greater cognitive burden (Mehri et al., 2026; Zhong et al., 2024; Maharana et al., 2024; Liu et al., 2026b). In Magentic-UI, this bottleneck appears as repeated confirmation requests for intermediate actions (e.g., flight search and price checking) and frequent re-elicitation of preferences (e.g., cabin class) across similar sessions. In contrast, agentic systems that learn from past interactions can adapt to evolving user preferences, thereby enabling more efficient human–agent communication.

Recent advances have explored how agentic systems can infer user preferences from interaction history (Yang et al., 2025; Mehri et al., 2026; Zhang et al., 2025; Westhäußer et al., 2025). However, preference inference alone does not necessarily translate into more efficient task execution unless the system can effectively communicate with users to incorporate unique human insight and strategic human oversight (Bansal et al., 2024; Goyal et al., 2024; Sterz et al., 2024). Recent studies have therefore begun to examine when agents should engage users during task execution (Dong et al., 2026; Zhou et al., 2025; Irons et al., 2025; Fuchs et al., 2024). Nonetheless, the absence of clear and easy intervention paths can still force users into cognitively demanding and time-consuming interactions (Wu et al., 2025; Simkute et al., 2025; Dutta et al., 2025). Efficient human–agent communication depends not only on *when to loop in the user*, but also on *how to enable effective intervention*.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

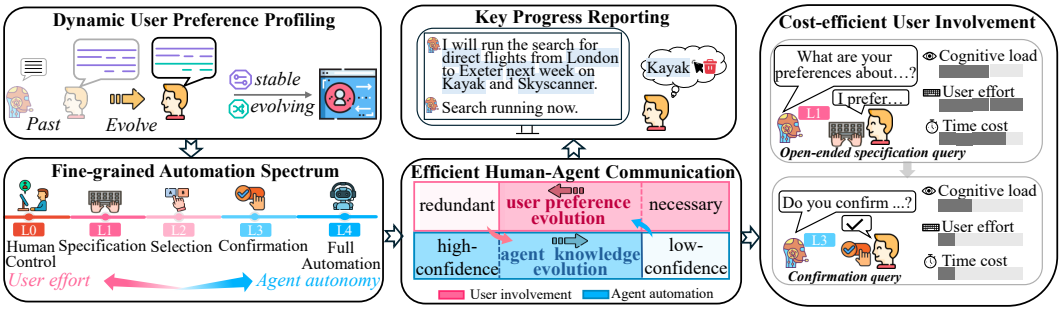


Figure 1: Overview of EASE. Dynamic user profiling and the fine-grained automation spectrum jointly guide adaptive human–agent communication, eliciting necessary user input in a cost-efficient manner. EASE further surfaces actionable workflow progress to support timely intervention.

Achieving such effective communication, however, requires navigating a complex interdependency between human–agent interaction patterns and the system's adaptation trajectory. Current systems often exhibit conversational inertia that anchors them to previous interaction patterns (Wan et al., 2026; Jiang et al., 2025; Tan et al., 2025). However, as user goals shift, preferences can also evolve (Gao et al., 2024; Sun et al., 2025; Son et al., 2024; Zhang et al., 2026). By acting on stale preferences, such systems can *over-automate* parts of the workflow and suppress necessary human–agent communication, leading to misalignment between agent behaviour and user intent (Kahn et al., 2024; Naik et al., 2025; Qian et al., 2025). An agentic system cannot adapt effectively if it relies only on past interactions while failing to account for evolving user preferences. Meanwhile, the continuous adaptation of agentic systems is constrained by *interaction-form rigidity*, preventing them from calibrating user involvement to the least burdensome interaction form sufficient to obtain the necessary input (Acikgoz et al., 2026; Wang et al., 2025). Moreover, efficient human–agent communication depends on timely human intervention, which in turn requires the system to reconcile transparency with simplicity. While providing details about agent behaviour can improve transparency, *over-communication* dilutes user attention and reduces the likelihood of catching critical information (Bansal et al., 2024; Herm, 2023; Sullivan & Weger, 2025; Rezwana & Ford, 2025).

To address these challenges, we introduce EASE, an efficient human–agent communication framework for evolving user preferences, which continuously adapts communication through a feedback loop: accumulated interactions refine the user profile, which in turn guides when and how the agent engages the user. At its core, as illustrated in Fig. 1, EASE moves beyond history-bound adaptation by enabling more flexible and granular communication control over when users are engaged, how user input is elicited, and what information is surfaced. Concretely, EASE builds a ***dynamic user profile*** by integrating user preference learning with online preference shift tracking, allowing the system to capture both stable long-term preferences and emerging preference changes. On top of this evolving profile, EASE organizes communication through a ***fine-grained automation spectrum*** with progressively lower interaction cost, ranging from *Human Control* and *Specification* to *Selection*, *Confirmation*, and *Full Automation*. By calibrating the level of agentic automation to the consistency of user preferences, this spectrum enables agile adaptation of communication strategies. To support timely intervention without fatiguing the user, EASE further incorporates a ***key progress reporting*** module, which selectively surfaces critical execution states while suppressing low-level operational details. By strategically coordinating user engagement, interaction form, and progress exposure, EASE balances agent autonomy with appropriate user involvement, reducing interaction burden while preserving opportunities for timely human intervention.

The main contributions of this work are as follows.

- We reveal a reciprocal loop in human–agent communication, where accumulated interactions improve communication efficiency, while subsequent user feedback drives the continual evolution of the agent's user model and communication strategy.
- We propose EASE, a novel framework that enables agentic systems to reduce redundant user involvement as interactions accumulate, remain responsive to evolving user preferences, and surface critical information for timely intervention.

- We develop a dimension-level adaptive communication mechanism that dynamically calibrates user involvement along a fine-grained automation spectrum as preferences evolve, enabling lower-effort elicitation through selection and confirmation when appropriate.
- We demonstrate that the synergy between evolving user profiles and adaptive communication strategies substantially reduces interaction overhead, yielding up to a 53.50% reduction in time cost under preference dynamics compared with the baselines.
- We conduct extensive experiments and ablation studies against strong baselines, analyzing the trade-off between user effort and agent autonomy in human–agent communication and showing that EASE reduces user-visible output volume by up to 73.50%.

2 Related work

Preference adaptation in agentic systems has received increasing attention, with recent studies exploring how accumulated interaction data can be used to infer user preferences and guide user engagement (Gomez et al., 2025). Some approaches integrate user profiles into long-term memory and create conditional delegation rules (Lai et al., 2022; Fuchs et al., 2023; Yang et al., 2025; Mehri et al., 2026; Zhang et al., 2025; Westhäußer et al., 2025) to determine the timing of user engagement. However, optimizing timing alone is insufficient, since agentic systems may still fail to incorporate critical human insight if user input is not elicited effectively. Some advanced frameworks attempt to address when and how to ask questions jointly (Acikgoz et al., 2026; Andukuri et al., 2024), but they largely focus on resolving ambiguity within the current session, without explicitly modelling how user preferences evolve over time. Reliance on stale user profiles can lead to over-automation and degraded decision accuracy when user preferences shift. Our work fills this gap by optimizing when and how agents engage users via dynamic user profiling and adaptive automation-level control.

Human–agent communication in agentic systems has supported varying degrees of user involvement, ranging from open-ended clarification and approval to autonomous execution (Mozannar et al., 2025; Feng et al., 2025). However, these mechanisms generally provide limited adaptation of the interaction form to evolving user preferences at the dimension level, leading to unnecessary user effort under stable preferences and insufficient re-engagement when preferences change. Moreover, effective human–agent communication depends not only on how user input is elicited, but also on how execution progress is exposed (Brachman et al., 2025; Yu et al., 2025). Existing systems often surface abundant operational details to improve transparency, yet the massive cognitive burden makes it hard for the user to identify the critical information for timely intervention (He et al., 2025; Treude & Kula, 2025; Muralidhar et al., 2025). Our work addresses these limitations through a fine-grained automation spectrum and key progress reporting, enabling more flexible elicitation and more efficient progress exposure for adaptive and communication-efficient agentic systems.

3 EASE

We present EASE, an adaptive communication framework for efficient human–agent interaction under evolving user preferences. As illustrated in Fig. 2, EASE consists of three layers: a data layer, an interaction layer, and a model layer. The data layer provides the adaptive foundation by tracking session states, maintaining a dynamic user profile, and abstracting raw execution outcomes into key progress updates. Built on this adaptive foundation, the interaction layer parses user intents, adapts automation levels according to preference stability, intervention strength, and task risk, and selects the appropriate interaction form among free-text specification, structured selection, confirmation, and autonomous execution. The model layer serves as the execution backbone, where language models and external tools carry out tasks and return structured outputs to the upper layers. Through this layered design, EASE incrementally evolves from interaction history, remains responsive to preference evolution, and calibrates communication strategies accordingly, enabling agent behaviour to align more closely with user intent while minimizing user effort.

3.1 Fine-grained automation spectrum

Given a task session indexed by n, EASE decomposes the task into a set of *interaction dimensions*, denoted by $\mathcal{D} = \{1, \ldots, d, \ldots, D\}$. An interaction dimension can correspond to a task parameter

(e.g., cabin class or number of passengers) or an execution decision (e.g., whether to initiate search or submit booking). For each dimension d, the system selects an automation level $\ell_i^{n,d} \in \mathcal{L} = \{L_0, L_1, L_2, L_3, L_4\}$ at interaction step i, where L_0 denotes *Human Control*, L_1 *Specification*, L_2 *Selection*, L_3 *Confirmation*, and L_4 *Full Automation*, with decreasing user effort along the spectrum, i.e., $\mathrm{Cost}(L_k) \geq \mathrm{Cost}(L_{k+1})$ for $k = 0, \ldots, 3$. Specifically, at L_0, the user either performs the step or overrides the agent. At L_1, the agent elicits the user's intent with an open-ended query (e.g., "Which search platforms do you prefer?"). At L_2, the agent narrows the decision space by presenting a small set of candidate options (e.g., "Which platforms should I search: Google Flights, Skyscanner, Kayak, or all of them?"). At L_3, the agent asks the user to confirm a proposed action or task parameter (e.g., "Confirm search on Skyscanner?"). At L_4, the agent proceeds autonomously without further user input. Appendix A provides clarification categories and examples.

EASE organizes user engagement along a cost-aware automation spectrum and dynamically calibrates the degree of human involvement at the dimension level according to preference stability. This design frames communication as an adaptive calibration process, enabling lower-effort interaction for stable preferences while re-engaging users as preferences evolve.

3.2 Dynamic User Preference Profile

EASE builds a dynamic user preference profile from accumulated interaction data to capture stable and evolving preferences. We denote the retained interaction history as $\mathcal{H} = \{\mathbf{H}^1, \ldots, \mathbf{H}^n, \ldots, \mathbf{H}^N\}$, where N indexes the current session within the history window. Each session contains records over a set of interaction dimensions, $\mathbf{H}^n = \{\mathbf{H}^{n,1}, \ldots, \mathbf{H}^{n,d}, \ldots, \mathbf{H}^{n,D}\}$, where $\mathbf{H}^{n,d} = \{h_1^{n,d}, \ldots, h_i^{n,d}, \ldots, h_{Z_{n,d}}^{n,d}\}$ is the sequence of actions on dimension d in session n. Each action is recorded as $h_i^{n,d} = (\ell_i^{n,d}, a_i^{n,d}, u_i^{n,d})$, where $\ell_i^{n,d}$ denotes the automation level, $a_i^{n,d}$ the agent output, and $u_i^{n,d}$ the user input. For each completed action, EASE extracts $\mathrm{Ans}(n, d, i)$, the finalized preference or task value for dimension d at step i in session n (e.g., `Economy` or `2 adults`). Session-specific requirements apply only to the current task and are not treated as persistent preference evidence unless explicitly indicated otherwise. When an autonomously inferred or reused value is exposed to the user, the absence of correction is treated as implicit confirmation of preference stability, whereas a correction indicates preference change.

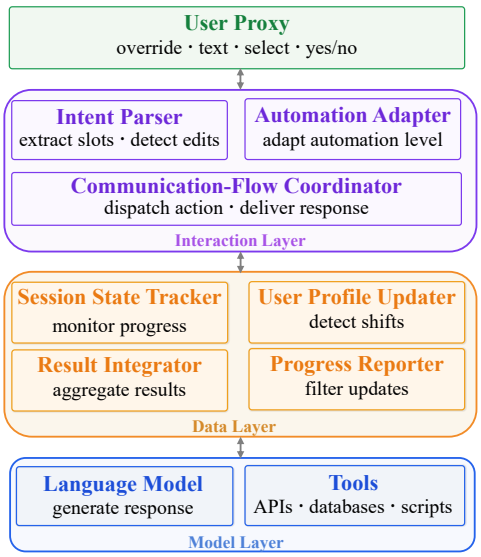


Figure 2: Overview of EASE.

For a dimension without a prior value, the system uses a task-specific default automation level $\ell_1^{1,d} = \ell_{\text{default}}^d$. For the current session N, let $z_d := Z_{N,d}$ denote the current action step on dimension d, for which the automation level $\ell_{z_d}^{N,d}$ is to be determined. EASE computes a stability score from both preference evolution *in the current session* and preference stability *across historical sessions*.

$$S(N, d, z_d) = \beta\, C_{\text{current}}(N, d, z_d) + (1 - \beta)\, H_{\text{cross}}(N, d, z_d), \tag{1}$$

where $\beta \in [0, 1]$ balances short-term preference dynamics and long-term behaviour consistency. $C_{\text{current}}(N, d, z_d)$ measures within-session coherence, with lower values indicating stronger shifts.

$$C_{\text{current}}(N, d, z_d) = \mathbb{I}(z_d \geq 3) \cdot \frac{1}{z_d - 2} \sum_{i=2}^{z_d - 1} \mathrm{Sim}(\mathrm{Ans}(N, d, i), \mathrm{Ans}(N, d, i-1)). \tag{2}$$

$H_{\text{cross}}(N, d, z_d)$ captures cross-session preference stability by comparing the most recently resolved value in the current session with the final resolved values from prior sessions.

$$H_{\text{cross}}(N, d, z_d) = \frac{\sum_{n=1}^{N-1} w(N-n)\,\text{Sim}(\text{Ans}(n, d, Z_{n,d}), \text{Ans}(N, d, z_d - 1))}{\sum_{n=1}^{N-1} w(N-n)}. \tag{3}$$

Here, $w(N-n)$ is a time-decay function that assigns larger weights to more recent sessions. We define $\text{Sim}(\cdot, \cdot)$ according to the value type. For categorical or numeric values, we use exact matching, i.e., $\text{Sim}(x, y) = 1$ if $x = y$, and 0 otherwise. For set-valued answers, we use Jaccard similarity, i.e., $\text{Sim}(x, y) = |x \cap y|/|x \cup y|$. For free-text values, we use cosine similarity, i.e., $\text{Sim}(x, y) = (\mathbf{v}_x^\top \mathbf{v}_y)/(\|\mathbf{v}_x\|\,\|\mathbf{v}_y\|)$, where $\mathbf{v}$ denotes the text embedding. The profile distills accumulated interaction data into dimension-level preference estimates, enabling EASE to evolve its communication strategy and guide subsequent automation-level adaptation.

3.3 Adaptive automation-level control

Building on the dynamic user profile, EASE treats automation-level control as an adaptive communication problem rather than a static mapping from interaction history to agent autonomy. The stability score $S(N, d, z_d)$ indicates whether accumulated interaction data supports a shift toward lower-effort interaction forms, but the next automation level must also reflect the current interaction context. In practice, active user interventions serve as real-time course correction signals that steer subsequent agent actions, while dimension risk constrains agent autonomy when erroneous actions may incur high costs. EASE therefore updates automation levels by jointly considering preference stability, intervention strength, and dimension risk.

Let $\delta(N, d, z_d - 1) \in \{0, 1\}$ indicate whether the preceding action on dimension d introduces human input relevant to the current automation-level update. Its computation is detailed in Appendix B. We define intervention strength as $\theta(N, d, z_d) = \delta(N, d, z_d - 1)\, g_{\text{itv}}(u^{N,d}_{z_d-1})$, where $g_{\text{itv}}(\cdot) \in [0, 1]$ maps the latest user input to a normalized course-correction score with respect to the current action or parameter. A smaller $\theta(N, d, z_d)$ indicates a minor refinement that can often be directly incorporated by agents without disrupting the current workflow, whereas a larger value indicates a stronger deviation from the current plan and therefore favours lower automation levels. To prevent over-automation on high-stakes decisions, each interaction dimension is assigned a risk score $\rho(d) \in [0, 1]$, where high-stakes dimensions such as booking confirmation and order placement receive larger values than lower-risk dimensions such as search engine selection or result sorting. For arithmetic updates, we represent each level L_k by its index $k \in \{0, 1, 2, 3, 4\}$. To combine these factors, EASE defines an adaptation score as

$$R(N, d, z_d) = \big(S(N, d, z_d) - \tau\big) - \lambda\,\theta(N, d, z_d) - \mu\,\rho(d)\,\ell^{N,d}_{z_d-1}, \tag{4}$$

where τ is a stability threshold, $\lambda > 0$ controls sensitivity to user intervention, and $\mu > 0$ is a risk-aversion coefficient. Positive scores encourage greater automation, whereas negative scores favour more human involvement. To preserve incremental evidence across interaction steps, EASE maintains a continuous internal automation state $q_{N,d,z_d} \in [0, 4]$. For each dimension, the state is initialized from its most recent available state, or from the index of ℓ^d_{default} if no prior state exists. It is updated as $q_{N,d,z_d} = \text{clip}(q_{N,d,z_d-1} + \eta R(N, d, z_d), 0, 4)$ and projected to the discrete automation level as

$$\ell^{N,d}_{z_d} = \Pi_{\mathcal{L}}(q_{N,d,z_d}), \tag{5}$$

where $\Pi_{\mathcal{L}}(q) = L_{\text{round}(q)}$, and $\eta > 0$ controls the adjustment rate. The continuous state accumulates sub-level changes, while the projected level determines the interaction form exposed to the user. Preference-stable, low-risk dimensions move toward lighter-weight interaction or full automation, whereas active interventions or higher risks shift them toward greater human involvement. Implementation settings and parameter sensitivity are provided in Appendix C.

3.4 Key progress reporting

Effective human–agent communication requires not only efficient incorporation of user input, but also selective exposure of execution progress so that users can easily identify misalignments and intervene. To this end, EASE distinguishes between two types of task states through $\kappa : \mathcal{D} \to \{0, 1\}$,

where $\kappa(d) = 0$ denotes a *condition state* and $\kappa(d) = 1$ denotes an *execution state*. Condition states encode task constraints or parameters, while execution states capture actionable workflow progress. Rather than reporting every parameter update or tool-use trace, EASE reports only decision-critical autonomous execution progress. Formally, the report trigger indicator is defined as $\mathrm{Trig}(n, d, i) = \mathbb{I}\big(\kappa(d) = 1\big) \cdot \mathbb{I}\big(\ell_i^{n,d} = L_4\big) \cdot \mathbb{I}\big(u_i^{n,d} = \mathrm{None}\big)$. Let $X^{(n)} = \{x_d \mid \kappa(d) = 0\}$ be the finalized condition-state vector for session n. When triggered, the report concatenates the execution-state phrase $g_{\mathrm{phs}}(d)$ with a concise summary of the finalized conditions $g_{\mathrm{con}}(X^{(n)})$:

$$r_i^{n,d} = g_{\mathrm{phs}}(d) \oplus g_{\mathrm{con}}(X^{(n)}). \tag{6}$$

For example, if $d = \mathit{SearchAction}$ and $X^{(n)} = \{\texttt{one-way}, \texttt{economy}, \texttt{2 passengers}, \texttt{London} \rightarrow \texttt{Paris}, \texttt{Skyscanner}\}$, EASE reports: "I will start searching for one-way economy flights for 2 passengers from London to Paris on Skyscanner." By filtering out low-level parameter updates and tool-use traces, EASE mitigates over-communication, where excessive updates dilute the user's attention. The selective reporting mechanism reduces cognitive burden, increasing the likelihood that the user can catch critical information and course-correct in a timely manner. Further details on progress-report generation are provided in Appendix D.

4 Environment and Workflow

4.1 Agentic Environment

We instantiate EASE as a task-oriented agent environment that supports both free-text input and lightweight user actions, enabling fine-grained calibration of agent autonomy through granular control over interaction forms. At the model layer, we use a compact local language model, `Qwen2.5-7B-Instruct` (Yang et al., 2024), as the execution backbone due to its efficient inference, long-context support, and stable instruction-following performance. Notably, EASE is model-agnostic, since its communication decisions are dynamically orchestrated by the upper-layer modules rather than hard-coded into the backbone model.

4.2 Communication Workflow

Algorithm 1 summarizes how EASE adapts communication and automation across interaction dimensions as user preferences evolve. At each interaction step, EASE treats communication as a routing problem, where each active interaction dimension is routed to human interaction, autonomous execution, or progress reporting. The system first identifies active dimensions and synchronizes the session state. For each active dimension, it then computes preference stability, intervention strength, and risk to update the automation level. If human involvement is required, the interaction coordinator selects an appropriate lightweight interaction form. After the requested user input is received, the updated session state re-enters the workflow, with an affirmative L_3 confirmation authorizing the pending action to resume execution. Dimensions not requiring human involvement are delegated directly to the model layer for autonomous execution. After execution, EASE updates user profile based on finalized dimension values, allowing future sessions to reuse stable patterns while remaining responsive to preference shifts. The reporting module selectively exposes workflow progress and decision-relevant outcomes to support timely user intervention.

5 Experiments

5.1 Experimental Setup

Task domains. We instantiate the agentic environment in two task domains. (1) *Flight Booking*. Condition dimensions include route, airports, number of passengers, cabin class, direct-only constraint, and search platforms, while execution dimensions include search initiation and booking submission. (2) *Meeting Scheduling*. Condition dimensions include attendee selection, meeting type, time slot, meeting mode, and location, while execution dimensions involve availability checking and invitation sending. Together, they capture core agentic challenges, such as multi-dimensional user elicitation and cross-session adaptation. Representative task databases are provided in Appendix E.

Algorithm 1 EASE: Efficient Human–Agent Communication Workflow

Require: user input u_t; session index N; task agent $\mathcal{A}$; history $\mathcal{H}$; interaction dimensions $\mathcal{D}$
Ensure: either ENGAGE(m), REPORT$(\mathcal{R}_t)$, or RESPOND(r)

$(\hat{\mathcal{D}}_t, \hat{\mathbf{x}}_t, \hat{\boldsymbol{\delta}}_t) \leftarrow$ INTENTPARSER(u_t) ▷ detect interventions and extract slots
SESSIONSTATETRACKER.UPDATE$(N, \hat{\mathbf{x}}_t, \hat{\boldsymbol{\delta}}_t)$ ▷ synchronize session state across layers
for each active dimension $d \in \hat{\mathcal{D}}_t$ **do**
  $z_d \leftarrow Z_{N,d} + 1$; $\ell_d \leftarrow$ ADAPTLEVEL$(N, d, z_d; \mathcal{H})$ ▷ update automation level
  if $\ell_d \in \{L_0, L_1, L_2, L_3\}$ **then**
    $m \leftarrow$ COORDINATEINTERACTION(d, ℓ_d, N) ▷ human–agent interaction
    return ENGAGE(m)
  end if
  $o_d \leftarrow$ MODELLAYER.EXECUTE$(\mathcal{A}, d, X^{(N)})$ ▷ automate agentic execution
  $r_d \leftarrow$ RESULTINTEGRATOR.UPDATE(d, o_d)
end for
PREFERENCEPROFILEUPDATER.COMMIT(N) ▷ update user profile with finalized values
$\mathcal{R}_t \leftarrow \{r_{z_d}^{N,d} \mid d \in \mathcal{D}, \mathrm{Trig}(N, d, z_d) = 1\}$
if $\mathcal{R}_t \neq \varnothing$ **then return** REPORT$(\mathcal{R}_t)$ ▷ report key progress
return RESPOND(INTEGRATEDRESULT(N))

Interaction settings. Our evaluation includes both simulated and human users. The user simulator is detailed in Appendix F. We consider three settings: (1) *No Prior History*, where the system has no previous records of the user. (2) *Stable Preferences*, where user preferences remain consistent across sessions. (3) *Changing Preferences*, where preferences drift over time. Preference and shift generation are detailed in Appendix G.

Baselines. The baselines span different communication and preference-adaptation strategies. (1) *Inference-Driven*: a model-based agentic system that implicitly decides whether to ask for clarification or proceed autonomously (Mozannar et al., 2025). (2) *Single-Turn*: a non-iterative interaction scheme that makes decisions solely based on the interactions during the current session (Cheng et al., 2022). (3) *Static-Profile*: a history-aware system that builds a user preference profile but does not adapt it online (Cai et al., 2025). (4) *DTAC*: a decision-theoretic method that optimizes clarification timing based on expected benefit and communication cost (Dong et al., 2026). (5) *CIPHER*: a preference-learning approach that improves outputs by leveraging latent user preferences from historical edits (Gao et al., 2024). All methods share the same interaction affordances, isolating the effects of automation-level adaptation, preference evolution modelling, and cost-aware elicitation.

Evaluation metrics. We evaluate performance along three dimensions. (1) *Interaction Cost*: measured by user action time using the keystroke-level model (Card et al., 1980) and input tokens. (2) *Cognitive Load*: measured by user-visible output volume. (3) *Adaptation Quality*: measured by automation-level growth and alignment with evolving user preferences.

5.2 MAIN RESULTS

Without requiring communication-policy training, EASE is easy to deploy and less dependent on task-specific data. Appendices H and I further characterize interaction behaviour through dialogue examples and user-input trends. Beyond controlled settings, Appendix J stress-tests EASE on public tool–agent–user benchmarks against request ambiguity, preference conflicts, tool-state changes, and consequential actions. Appendix K tests robustness to user/profile errors and over-automation.

EASE reduces interaction overhead through lightweight elicitation and key progress updates. As shown in Fig. 3, by replacing free-text input with selection and confirmation while filtering out low-level execution details, EASE reduces user action time and user-visible output volume by up to 23.77% and 73.50%, respectively. Unlike inference-driven, DTAC, and CIPHER baselines, which rely heavily on free-text interaction or verbose procedural reports, EASE streamlines user elicitation and progress reporting through adaptive communication control, shifting user attention from redundant clarification and procedural monitoring to critical decisions and strategic oversight.

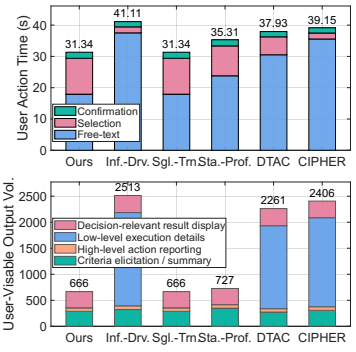


Figure 3: Interaction cost under no prior history.

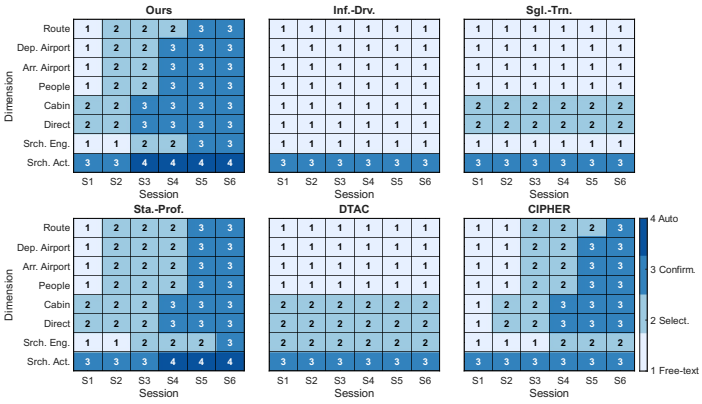
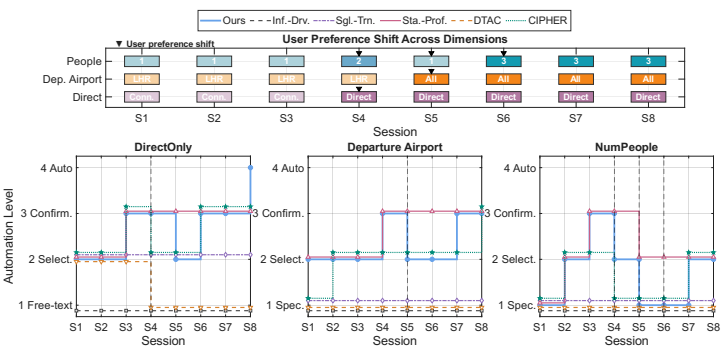
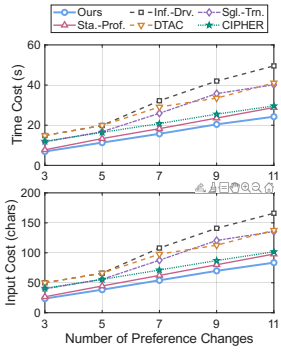


Figure 4: Automation-level evolution under stable preferences.



(a) Automation-level adaptation.



(b) Interaction cost.

Figure 6: Automation-level adaptation and interaction cost under evolving user preferences.

Fine-grained automation spectrum enables sustained efficiency gains under stable preferences. As shown in Fig. 4, EASE progressively elevates automation levels over repeated sessions, reaching up to $2.02\times$ the baselines' average automation level. This growth stems from its ability to translate accumulated confidence into a structured automation-level escalation path. As shown in Fig. 5, EASE reduces user action time by up to 72.22% by the sixth session. In contrast, baselines such as DTAC and CIPHER stagnate as they fail to translate consistent user behaviour into cost-aware automation-level adaptation. Overall, communication efficiency comes not from removing the user entirely, but from progressively reallocating user effort to the moments where human input is most valuable.

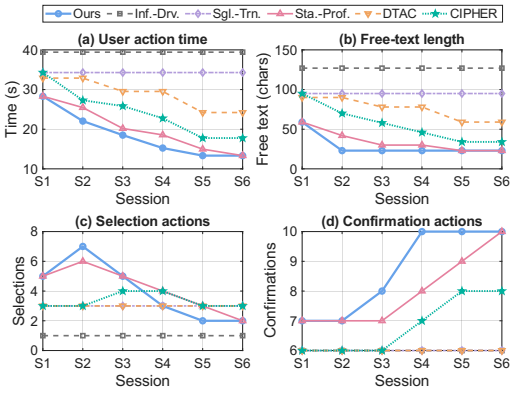


Figure 5: Interaction under stable preferences.

Adaptive automation control is essential under evolving user preferences. When user preferences evolve, conversational inertia can lead to over-automation, while excessive clarification increases communication overhead. As shown in Fig. 6a, EASE resolves this tension through adaptive automation control. It downgrades affected dimensions, elicits necessary input, and re-escalates automation once preferences stabilize. Fig. 6b shows that EASE reduces shift-handling time cost by up to 53.50% and user input cost by up to 42.42% compared with the baselines, which lack a unified mechanism to track preference evolution and recalibrate automation levels accordingly.

5.3 Ablation study

For component ablations, only the designated EASE component is removed, while the remaining system components and evaluation settings are kept unchanged. Table 1 shows that the gains of EASE reflect the joint contributions of its four components. Without a history-based user profile, the system cannot exploit behavioural stability across sessions, resulting in only a marginal 0.84% time reduction under stable preferences. History-based profiling identifies recurring user behaviour patterns, which the fine-grained automation spectrum converts into lower-effort interaction schemes. Removing this spectrum lowers automation-level growth by 10.87 percentage points and time reduction by 9.28 percentage points. Tracking preference evolution is critical for maintaining accurate execution. Without it, the system tends to over-automate, increasing shift-handling time by 67.29% as the user must intervene to correct the agent. In contrast, EASE achieves 84.62% adaptation alignment, reduces shift-handling time by 19.85%, and further improves communication efficiency, reducing user-visible output to 248 tokens, 15.93% below the second-best ablated variant.

Table 1: Ablation results across different interaction settings. Bold/underlined: best/second-best results.

Variants	No Hist.		Stable Pref.		Pref. Drift	
	UVO ↓	Time ↓	Auto. ↑	Red. ↑	Align. ↑	Sw. ↓
w/o Hist.	580	35.9	0.52	0.84	38.46	33.6
w/o Shift.	<u>295</u>	<u>29.4</u>	<u>28.80</u>	<u>49.10</u>	23.08	35.8
w/o FG Auto.	338	31.6	22.40	45.57	61.54	28.0
w/o KPR	970	32.4	19.93	37.96	<u>76.92</u>	<u>26.7</u>
EASE	**248**	**26.8**	**33.27**	**54.85**	**84.62**	**21.4**
Improv.	15.93%	8.84%	4.47 pp	5.75 pp	7.70 pp	19.85%

Note: Hist., Shift, FG Auto., and KPR denote the historical profile, preference-shift tracking, fine-grained automation spectrum, and key progress reporting, respectively. No Hist. denotes the absence of prior-session records, whereas w/o Hist. further disables history-based profiling for accumulating and reusing interaction evidence within the ongoing session. UVO: user-visible output volume, measured by output tokens presented to the user; Auto.: average automation-level growth; Red.: time reduction from the first session; Sw.: average shift-handling time; Align.: smaller-to-larger ratio between preference shifts and automation-level adjustments. Auto., Red., and Align. are in percentages. Improv. is computed over the second-best variant.

5.4 Human evaluation

Table 2 reports a real-user within-subject pilot with three participants under masked method identities. Under stable preferences, EASE improves all NASA-TLX dimensions by converting learned preferences into low-cost interaction affordances. Under preference drift, EASE selectively re-engages users, avoiding stale automation and unnecessary clarification. Together, the results suggest that EASE improves communication not by maximizing autonomy or merely providing additional interaction affordances, but by continuously calibrating the form and degree of user involvement to evolving contexts.

Table 2: Human evaluation of perceived workload and task performance. Bold/underlined: best/second-best results.

Method	**Stable Pref.**						**Pref. Drift**	
	MD ↓	**PD ↓**	**TD ↓**	**OP ↑**	**EF ↓**	**FR ↓**	**Time (s) ↓**	**Success ↑**
Single-Turn	50.0	42.3	46.7	73.0	48.3	43.3	58.6	75.0
Static-Profile	<u>35.0</u>	<u>27.0</u>	<u>31.7</u>	<u>80.0</u>	<u>27.3</u>	<u>28.3</u>	<u>32.5</u>	50.0
Inference-Driven	55.0	38.6	51.7	71.0	53.3	48.3	56.8	72.2
DTAC	46.7	30.0	43.3	75.0	45.0	40.0	39.1	77.7
CIPHER	40.0	28.0	36.7	78.7	38.3	33.3	35.1	<u>80.6</u>
EASE	**30.0**	**22.3**	**26.7**	**87.0**	**20.0**	**23.3**	**27.9**	**94.4**
Improv.	**14.3%**	**17.4%**	**15.8%**	**+7.0 pp**	**26.7%**	**17.7%**	**14.2%**	**+13.9 pp**

Note: For **Stable Pref.**, NASA-TLX scores (Liu et al., 2026a) were collected after all sessions and averaged over three participants. MD, PD, TD, OP, EF, and FR denote mental demand, physical demand, temporal demand, overall performance, effort, and frustration, respectively. For **Pref. Drift**, user action time and task success were recorded in every session and averaged across all sessions and participants.

6 Conclusion

We studied efficient human–agent communication under evolving user preferences, where agents must incorporate necessary user involvement without imposing redundant user effort. Existing systems remain constrained by rigid interaction policies, stale preference reuse, and verbose operational traces, leading to unnecessary user effort, over-automation under preference shifts, and information overload. To address these limitations, we proposed EASE, which combines dynamic user preference profiling, fine-grained automation-level control, and key progress reporting to adapt communication as user preferences evolve. Across no-history, stable-preference, and preference-drift settings, EASE consistently reduces interaction overhead, achieving up to a 53.50% reduction in time cost and a 73.50% reduction in user-visible output volume compared with the baselines. These results demonstrate that efficient human–agent communication is achieved not by maximizing autonomy alone, but by dynamically calibrating when, how, and to what extent users should be engaged.

AI USE STATEMENT

Generative AI tools were used to support the linguistic refinement of this manuscript, including sentence rephrasing, grammar checking, and improving the overall coherence and readability of the text. They were not used to formulate the research ideas, develop the methodology, design the experiments, or conduct the scientific analysis. All technical content, results, and conclusions were produced and verified by the authors, who take full responsibility for the final manuscript.

ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics. The human-user evaluation was conducted entirely in controlled, simulated task environments. No real bookings, payments, reservations, invitations, or other real-world transactions were executed. Task data were constructed for evaluation purposes, with contact records anonymized, and only aggregate human-evaluation results are reported. These measures were taken to minimize privacy risks and prevent the evaluated agentic actions from causing real-world consequences.

REPRODUCIBILITY STATEMENT

We have taken several steps to support the reproducibility of our experimental results. The paper and appendices provide the implementation and parameter settings, task databases, user-simulation and preference-generation procedures, as well as the evaluation and stress-test protocols. All methods are evaluated using the same task instances, preference trajectories, and user-simulation random seeds to ensure controlled and consistent comparisons.

REFERENCES

Mohamad Abou Ali, Fadi Dornaika, and Jinan Charafeddine. Agentic ai: a comprehensive survey of architectures, applications, and future directions. *Artificial Intelligence Review*, 59(1):11, 2025.

Emre Can Acikgoz, Jinoh Oh, Joo Hyuk Jeon, Jie Hao, Heng Ji, Dilek Hakkani-Tur, Gokhan Tur, Xiang Li, Chengyuan Ma, and Xing Fan. MAC: A multi-agent framework for interactive user clarification in multi-turn conversations. In Giuseppe Riccardi, Seyed Mahed Mousavi, Maria Ines Torres, Koichiro Yoshino, Zoraida Callejas, Shammur Absar Chowdhury, Yun-Nung Chen, Frederic Bechet, Joakim Gustafson, Géraldine Damnati, Alex Papangelis, Luis Fernando D'Haro, John Mendonça, Raffaella Bernardi, Dilek Hakkani-Tur, Giuseppe "Pino" Di Fabbrizio, Tatsuya Kawahara, Firoj Alam, Gokhan Tur, and Michael Johnston (eds.), *Proceedings of the 16th International Workshop on Spoken Dialogue System Technology*, pp. 1–17, Trento, Italy, February 2026. Association for Computational Linguistics. URL https://aclanthology.org/2026.iwsds-1.1/.

Taha Aksu and Nancy Chen. Granular change accuracy: A more accurate performance metric for dialogue state tracking. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pp. 7939–7948, 2024.

Chinmaya Andukuri, Jan-Philipp Fränken, Tobias Gerstenberg, and Noah D Goodman. Star-gate: Teaching language models to ask clarifying questions. *arXiv preprint arXiv:2403.19154*, 2024.

Gagan Bansal, Jennifer Wortman Vaughan, Saleema Amershi, Eric Horvitz, Adam Fourney, Hussein Mozannar, Victor Dibia, and Daniel S Weld. Challenges in human-agent communication. *arXiv preprint arXiv:2412.10380*, 2024.

Han Bao, Zheyuan Zhang, Pengcheng Jing, Zhengqing Yuan, Kaiwen Shi, and Yanfang Ye. Driftbench: Diagnosing cooperative breakdowns in llm agents under input faults via multi-turn interaction. *arXiv preprint arXiv:2602.02455*, 2026.

Michelle Brachman, Siya Kunde, Sarah Miller, Ana Fucs, Samantha Dempsey, Jamie Jabbour, and Werner Geyer. Building appropriate mental models: What users know and want to know

about an agentic ai chatbot. In *Proceedings of the 30th International Conference on Intelligent User Interfaces*, IUI '25, pp. 247–264, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400713064. doi: 10.1145/3708359.3712071. URL https://doi.org/10.1145/3708359.3712071.

Hongru Cai, Yongqi Li, Wenjie Wang, Fengbin Zhu, Xiaoyu Shen, Wenjie Li, and Tat-Seng Chua. Large language models empowered personalized web agents. In *Proceedings of the ACM on Web Conference 2025*, pp. 198–215, 2025.

Stuart K. Card, Thomas P. Moran, and Allen Newell. The keystroke-level model for user performance time with interactive systems. *Commun. ACM*, 23(7):396–410, July 1980. ISSN 0001-0782. doi: 10.1145/358886.358895. URL https://doi.org/10.1145/358886.358895.

Ruijia Cheng, Alison Smith-Renner, Ke Zhang, Joel Tetreault, and Alejandro Jaimes-Larrarte. Mapping the design space of human-ai interaction in text summarization. In *Proceedings of the 2022 conference of the north American chapter of the Association for Computational Linguistics: Human language technologies*, pp. 431–455, 2022.

Alexandru Coca, Mark Gaynor, Zhenxing Zhang, Jianpeng Cheng, Bo-Hsiang Tseng, Pete Boothroyd, Héctor Martinez Alonso, Diarmuid Ó Séaghdha, and Anders Johannsen. Aspera: A simulated environment to evaluate planning for complex action execution. *arXiv preprint arXiv:2507.15501*, 2025.

Xiaoyu Dong, Yujie Feng, Zexin Lu, Guangyuan Shi, and Xiao-Ming Wu. Zero-shot cross-domain dialogue state tracking via context-aware auto-prompting and instruction-following contrastive decoding. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8527–8540, 2024.

Yijiang River Dong, Tiancheng Hu, Zheng Hui, Caiqi Zhang, Ivan Vulić, Andreea Bobu, and Nigel Collier. When should AI ask: Decision-theoretic adaptive communication for LLM agents, 2026. URL https://openreview.net/forum?id=idXrPgNmqx.

Prachi Dutta, Poonampreet K Josan, Raymond KW Wong, Bonnie J Dunbar, Ana Diaz-Artiles, and Daniel Selva. Effects of explanations and accuracy on human performance and trust in ai-assisted anomaly diagnosis tasks. *Journal of Cognitive Engineering and Decision Making*, 19(4):453–473, 2025.

Jinyuan Fang, Yanwen Peng, Xi Zhang, Yingxu Wang, Xinhao Yi, Guibin Zhang, Yi Xu, Bin Wu, Siwei Liu, Zihao Li, et al. A comprehensive survey of self-evolving ai agents: A new paradigm bridging foundation models and lifelong agentic systems. *arXiv preprint arXiv:2508.07407*, 2025.

Kevin J Feng, David W McDonald, and Amy X Zhang. Levels of autonomy for ai agents. *arXiv preprint arXiv:2506.12469*, 2025.

Andrew Fuchs, Andrea Passarella, and Marco Conti. Optimizing delegation between human and ai collaborative agents. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 245–260. Springer, 2023.

Andrew Fuchs, Andrea Passarella, and Marco Conti. Optimizing delegation in collaborative human-ai hybrid teams. *ACM Transactions on Autonomous and Adaptive Systems*, 19(4):1–33, 2024.

Ge Gao, Alexey Taymanov, Eduardo Salinas, Paul Mineiro, and Dipendra Misra. Aligning LLM agents by learning latent preference from user edits. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=DlYNGpCuwa.

Catalina Gomez, Sue Min Cho, Shichang Ke, Chien-Ming Huang, and Mathias Unberath. Human-ai collaboration is not very collaborative yet: A taxonomy of interaction patterns in ai-assisted decision making from a systematic review. *Frontiers in Computer Science*, 6:1521066, 2025.

Nitesh Goyal, Minsuk Chang, and Michael Terry. Designing for human-agent alignment: Understanding what humans want from their agents. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, CHI EA '24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400703317. doi: 10.1145/3613905.3650948. URL https://doi.org/10.1145/3613905.3650948.

Gaole He, Gianluca Demartini, and Ujwal Gadiraju. Plan-then-execute: An empirical study of user trust and team performance when using llm agents as a daily assistant. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400713941. doi: 10.1145/3706598.3713218. URL https://doi.org/10.1145/3706598.3713218.

Lukas-Valentin Herm. Impact of explainable ai on cognitive load: Insights from an empirical study. *arXiv preprint arXiv:2304.08861*, 2023.

Jessica Irons, Patrick Cooper, Melanie McGrath, Shahroz Tariq, and Andreas Duenser. Towards a criteria-based approach to selecting human-ai interaction mode. *Human Factors and Ergonomics in Manufacturing & Service Industries*, 35(4):e70022, 2025.

Bowen Jiang, Zhuoqun Hao, Young-Min Cho, Bryan Li, Yuan Yuan, Sihao Chen, Lyle Ungar, Camillo J Taylor, and Dan Roth. Know me, respond to me: Benchmarking llms for dynamic user profiling and personalized responses at scale. *arXiv preprint arXiv:2504.14225*, 2025.

Lauren Kahn, Emelia S Probasco, and Ronnie Kinoshita. Ai safety and automation bias. *Center for Security and Emerging Technology*, 2024.

Vivian Lai, Samuel Carton, Rajat Bhatnagar, Q. Vera Liao, Yunfeng Zhang, and Chenhao Tan. Human-ai collaboration via conditional delegation: A case study of content moderation. CHI '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391573. doi: 10.1145/3491102.3501999. URL https://doi.org/10.1145/3491102.3501999.

Xingyan Liu, Fei Meng Da Zhao, and Fei Qi. Design: A user study on cognitive load and usability. In *HCI International 2026 Posters: 28th International Conference on Human-Computer Interaction, HCII 2026, Montreal, QC, Canada, July 26–31, 2026, Proceedings, Part VI*, pp. 41. Springer Nature, 2026a.

Yimeng Liu, Misha Sra, Jeevana Priya Inala, and Chenglong Wang. Reuseit: Synthesizing reusable ai agent workflows for web automation. In *Proceedings of the 31st International Conference on Intelligent User Interfaces*, IUI '26, pp. 885–908. ACM, March 2026b. doi: 10.1145/3742413.3789083. URL http://dx.doi.org/10.1145/3742413.3789083.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of LLM agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13851–13870, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.747. URL https://aclanthology.org/2024.acl-long.747/.

Shuhaib Mehri, Priyanka Kargupta, Tal August, and Dilek Hakkani-Tür. Learning user preferences through interaction for long-term collaboration. *arXiv preprint arXiv:2601.02702*, 2026.

Hussein Mozannar, Gagan Bansal, Cheng Tan, Adam Fourney, Victor Dibia, Jingya Chen, Jack Gerrits, Tyler Payne, Matheus Kunzler Maldaner, Madeleine Grunde-McLaughlin, et al. Magentic-ui: Towards human-in-the-loop agentic systems. *arXiv preprint arXiv:2507.22358*, 2025.

Deepa Muralidhar, Rafik Belloum, and Ashwin Ashok. Operationalizing selective transparency using progressive disclosure in artificial intelligence clinical diagnosis systems. *International Journal of Human-Computer Studies*, pp. 103591, 2025.

Akshat Naik, Patrick Quinn, Guillermo Bosch, Emma Gouné, Francisco Javier Campos Zabala, Jason Ross Brown, and Edward James Young. Agentmisalignment: Measuring the propensity for misaligned behaviour in llm-based agents. *arXiv preprint arXiv:2506.04018*, 2025.

Yaoyao Qian, Jindan Huang, Yuanli Wang, Simon Yu, Kyrie Zhixuan Zhou, Jiayuan Mao, Mingfu Liang, and Hanhan Zhou. When to act, when to wait: Modeling the intent-action alignment problem in dialogue. *arXiv preprint arXiv:2506.01881*, 2025.

Jeba Rezwana and Corey Ford. Human-centered ai communication in co-creativity: An initial framework and insights. In *Proceedings of the 2025 Conference on Creativity and Cognition*, C&C '25, pp. 651–665, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400712890. doi: 10.1145/3698061.3726932. URL https://doi.org/10.1145/3698061.3726932.

Auste Simkute, Lev Tankelevitch, Viktor Kewenig, Ava Elizabeth Scott, Abigail Sellen, and Sean Rintel. Ironies of generative ai: understanding and mitigating productivity loss in human-ai interaction. *International Journal of Human–Computer Interaction*, 41(5):2898–2919, 2025.

Seongho Son, William Bankes, Sayak Ray Chowdhury, Brooks Paige, and Ilija Bogunovic. Right now, wrong then: Non-stationary direct preference optimization under preference drift. *arXiv preprint arXiv:2407.18676*, 2024.

Sarah Sterz, Kevin Baum, Sebastian Biewer, Holger Hermanns, Anne Lauber-Rönsberg, Philip Meinel, and Markus Langer. On the quest for effectiveness in human oversight: Interdisciplinary perspectives. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '24, pp. 2495–2507, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704505. doi: 10.1145/3630106.3659051. URL https://doi.org/10.1145/3630106.3659051.

Virginia Sullivan and Kristin Weger. Transparency and explainability in ai-assisted decision making: Effects on trust, perceived reliability, confidence, and ease of understanding. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 69, pp. 1827–1833. SAGE Publications Sage CA: Los Angeles, CA, 2025.

Haoran Sun, Zekun Zhang, and Shaoning Zeng. Preference-aware memory update for long-term llm agents. *arXiv preprint arXiv:2510.09720*, 2025.

Zhen Tan, Jun Yan, I-Hung Hsu, Rujun Han, Zifeng Wang, Long Le, Yiwen Song, Yanfei Chen, Hamid Palangi, George Lee, et al. In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8416–8439, 2025.

Christoph Treude and Raula Gaikovina Kula. Interacting with ai reasoning models: Harnessing" thoughts" for ai-driven software engineering. *arXiv preprint arXiv:2503.00483*, 2025.

Yang Wan, Zheng Cao, Zhenhao Zhang, Zhengwen Zeng, Shuheng Shen, Changhua Meng, and Linchao Zhu. Mitigating conversational inertia in multi-turn agents. *arXiv preprint arXiv:2602.03664*, 2026.

Ante Wang, Yujie Lin, Jingyao Liu, Suhang Wu, Hao Liu, Xinyan Xiao, and Jinsong Su. Beyond passive critical thinking: Fostering proactive questioning to enhance human-ai collaboration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 33350–33358, 2026.

Wenxuan Wang, Shi Juluan, Zixuan Ling, Yuk-Kit Chan, Chaozheng Wang, Cheryl Lee, Youliang Yuan, Jen-tse Huang, Wenxiang Jiao, and Michael R. Lyu. Learning to ask: When LLM agents meet unclear instruction. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 21773–21784, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1104. URL https://aclanthology.org/2025.emnlp-main.1104/.

Rebecca Westhäußer, Wolfgang Minker, and Sebatian Zepf. Enabling personalized long-term interactions in llm-based agents through persistent memory and user profiles. *arXiv preprint arXiv:2510.07925*, 2025.

Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. Collabllm: From passive responders to active collaborators. *arXiv preprint arXiv:2502.00640*, 2025.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

Cheng Yang, Xuemeng Yang, Licheng Wen, Daocheng Fu, Jianbiao Mei, Rong Wu, Pinlong Cai, Yufan Shen, Nianchen Deng, Botian Shi, et al. Learning on the job: An experience-driven self-evolving agent for long-horizon tasks. *arXiv preprint arXiv:2510.08002*, 2025.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains, 2024. URL https://arxiv.org/abs/2406.12045.

Da Yin, Faeze Brahman, Abhilasha Ravichander, Khyathi Chandu, Kai-Wei Chang, Yejin Choi, and Bill Yuchen Lin. Agent lumos: Unified and modular training for open-source language agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12380–12403, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.670. URL https://aclanthology.org/2024.acl-long.670/.

Ailin Yu, Zehui Jin, Yan Zhu, Yue He, Tao Yang, and Luping Wang. Exploring the impact of process transparency on user experience in ai design agents. In *Companion Publication of the 2025 Conference on Computer-Supported Cooperative Work and Social Computing*, CSCW Companion '25, pp. 379–384, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400714801. doi: 10.1145/3715070.3749256. URL https://doi.org/10.1145/3715070.3749256.

Han Zhang, Lin Gui, Yu Lei, Yuanzhao Zhai, Yehong Zhang, Zhuo Zhang, Yulan He, Hui Wang, Yue Yu, Kam-Fai Wong, et al. Copr: Continual human preference learning via optimal policy regularization. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 5377–5398, 2025.

Ruichen Zhang, Guangyuan Liu, Yinqiu Liu, Changyuan Zhao, Jiacheng Wang, Yunting Xu, Dusit Niyato, Jiawen Kang, Yonghui Li, Shiwen Mao, et al. Toward edge general intelligence with agentic ai and agentification: Concepts, technologies, and future directions. *IEEE Communications Surveys & Tutorials*, 28:4285–4318, 2026.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19724–19731, Mar. 2024. doi: 10.1609/aaai.v38i17.29946. URL https://ojs.aaai.org/index.php/AAAI/article/view/29946.

Jieyu Zhou, Aryan Roy, Sneh Gupta, Daniel Weitekamp, and Christopher J MacLellan. When should users check? a decision-theoretic model of confirmation frequency in multi-step ai agent tasks. *arXiv preprint arXiv:2510.05307*, 2025.

Table 3: Clarification categories in EASE. Instead of always using open-ended clarification, EASE adapts the communication form according to the degree of uncertainty, preference stability, intervention signal, and task risk.

Scope	Clarification Category	Description / Example
General	Parameter underspecification	Required task information is missing, e.g., "Book me a flight" without destination or date; "Arrange a meeting" without attendee or time.
	Value ambiguity	The user provides an imprecise value, e.g., "next week", "afternoon", or "a short meeting", which requires grounding into an executable value.
	Entity disambiguation	A mentioned entity matches multiple candidates, e.g., multiple contacts named "Alex", or multiple airports in the same city.
	Confirmation of inferred preference	The system infers a likely value from history, e.g., usual cabin class, usual meeting duration, or usual meeting mode.
	Preference shift / correction	The user revises a previously stable preference, e.g., "Actually, connections are acceptable" or "Make it in person instead."
	High-risk execution decision	The next step has irreversible or high-impact consequences, e.g., booking submission or invitation sending.
Flight	Route and airport ambiguity	The route or airport is incomplete or ambiguous, e.g., London may refer to multiple departure airports.
	Travel preference elicitation	The system needs cabin class, passenger count, direct-only preference, or search-platform preference.
	Search-platform selection	The user may prefer all platforms or a subset such as Google Flights, Skyscanner, or Kayak.
	Booking confirmation	The user selects a flight and the agent must confirm price, route, source platform, and passenger count before booking.
Meeting	Contact ambiguity	A short name maps to multiple staff records, e.g., "Alex" may refer to different roles or departments.
	Meeting preference elicitation	The system resolves meeting type, time slot, duration, mode, location, and agenda requirement.
	Location selection	The meeting can be online, in an office, or in a meeting room, depending on mode and attendee preferences.
	Invitation confirmation	Before sending the invitation, the system summarizes attendee, subject, date, time, duration, mode, location, and agenda.

A Clarification categories and examples

Clarification in EASE is not treated as a single open-ended interaction mode. Instead, the system distinguishes different sources of uncertainty and maps them to different communication forms along the automation spectrum. Table 3 summarizes the main clarification categories used in our flight-booking and meeting-scheduling environments. It shows that the human–agent communication problem under evolving user preferences is more than deciding whether to ask a clarification question. Some uncertainties require open-ended specification, such as missing destinations or meeting attendees, while others can be resolved through lower-cost interaction forms such as selection or confirmation. As interaction history accumulates, EASE increasingly replaces repeated specification with selection, confirmation, or autonomous execution on stable and low-risk dimensions. When the user introduces a correction or when the dimension is high-risk, the system shifts back toward more explicit user involvement. This design allows EASE to incorporate necessary human input while avoiding unnecessary clarification.

Table 4: Main implementation and evaluation settings.

Setting	Value	Setting	Value
Backbone model	Qwen2.5-7B-Instruct	History window W	8 sessions
Core parameters	$\beta = 0.5,\ \eta = 1.0$	(τ, λ, μ)	$(0.60,\ 0.60,\ 0.50)$
Flight default level	L1: route, airports, passengers, search engines; L2: cabin, direct-only; L3: search action	Meeting default level	L1: contact, type, time, duration, mode, location; L2: agenda; L3: invitation sending
Consequential-action safeguard	Booking submission requires L3 confirmation	Consequential-action safeguard	Invitation sending requires L3 confirmation
Controlled evaluation	6 sessions per setting	Stress-test evaluation	4 sessions/task: 3 history + 1 evaluation

B Computation of the dimension-relevance indicator

The indicator $\delta(N, d, z_d - 1)$ determines whether the user input in the preceding action record on dimension d should affect the current automation-level update. Following active-slot recognition in dialogue state tracking (Aksu & Chen, 2024; Dong et al., 2024), EASE uses the intent parser to identify the semantic operation introduced by this input.

Recall that the preceding action record on dimension d is $h^{N,d}_{z_d-1} = \left(\ell^{N,d}_{z_d-1}, a^{N,d}_{z_d-1}, u^{N,d}_{z_d-1}\right)$. Using the user input $u^{N,d}_{z_d-1}$, the corresponding agent output $a^{N,d}_{z_d-1}$, and the retained interaction history $\mathcal{H}$, the intent parser predicts

$$o^{N,d}_{z_d-1} = g_{\text{op}}\left(u^{N,d}_{z_d-1}, a^{N,d}_{z_d-1}, \mathcal{H}\right), \qquad o^{N,d}_{z_d-1} \in \mathcal{O}, \tag{7}$$

where only interaction records available before the current automation-level update are considered. Here, g_{op} denotes the dimension-level semantic classifier implemented by the intent parser. It maps the user input and agent output in the preceding action record, together with the retained interaction history, to an operation in

$$\mathcal{O} = \{\textsc{Carryover}, \textsc{Inform}, \textsc{Revise}, \textsc{Affirm}, \textsc{Reject}\}. \tag{8}$$

CARRYOVER indicates that no dimension-relevant user input is identified in the preceding action. INFORM provides a value for the dimension, while REVISE modifies a previously proposed or retained value. AFFIRM and REJECT indicate acceptance and rejection of the corresponding agent proposal, respectively. The dimension-relevance indicator is then computed as

$$\delta(N, d, z_d - 1) = \mathbb{I}\left[o^{N,d}_{z_d-1} \neq \textsc{Carryover}\right]. \tag{9}$$

Thus, $\delta(N, d, z_d-1)$ controls whether the preceding user input contributes to the current automation-level update for dimension d. Conditioned on the corresponding agent output and retained interaction history, the intent parser associates the input with the relevant dimension even when the dimension itself is not explicitly stated. For example, let d = SEARCHPLATFORM and suppose that $u^{N,d}_{z_d-1}$ = "Actually search only on Skyscanner." The intent parser classifies the input as $o^{N,d}_{z_d-1}$ = REVISE, and therefore $\delta(N, \textsc{SearchPlatform}, z_d - 1) = 1$. The indicator determines the relevance of the input to the dimension-level update, whereas $g_{\text{itv}}(\cdot)$ in Section 3.3 quantifies the extent to which the input calls for revision of the proposed agent action or current task parameter.

C Implementation Settings and Parameter Analysis

Table 4 summarizes the main implementation and evaluation settings used in our experiments. Unless otherwise specified, these settings are shared across the evaluated methods, while the sensitivity of the key EASE parameters is examined below.

We analyze the sensitivity of EASE to five key parameters: β, τ, λ, μ, and η. Table 5 shows that, under frequent preference changes, smaller β and η lead to lower user action time, reflecting less

Table 5: Sensitivity to β and η under observed preference stability.

β	η	$\bar{S} \in [0.25, 0.40]$	$\bar{S} \in [0.45, 0.60]$	$\bar{S} \in [0.75, 0.90]$
0.2	0.5	**27.4**	24.9	22.6
0.2	1.0	29.1	23.8	20.7
0.2	1.5	31.8	25.6	19.8
0.5	0.5	28.6	23.6	21.4
0.5	1.0	30.5	**22.7**	19.5
0.5	1.5	33.2	24.1	18.9
0.8	0.5	31.7	24.4	20.8
0.8	1.0	34.3	23.5	18.7
0.8	1.5	37.6	24.8	**17.9**

Note: Each value reports the average user action time (s) per session, averaged over the flight-booking and meeting-scheduling domains. $\bar{S}$ denotes the observed preference-stability score interval.

Table 6: Joint sensitivity to τ, λ, and μ across task domains.

Task	(τ, λ, μ)	Time (s) $\downarrow$	Auto. Exec. (%)	User Corr. (%) $\downarrow$
Meeting	$(0.50, 0.30, 0.20)$	**23.6**	71.4	3.2
	$(0.60, 0.60, 0.50)$	25.9	63.8	1.7
	$(0.75, 0.90, 0.90)$	32.7	49.5	**0.8**
Flight	$(0.50, 0.30, 0.20)$	**24.2**	76.1	8.6
	$(0.60, 0.60, 0.50)$	27.0	60.4	2.9
	$(0.75, 0.90, 0.90)$	33.4	43.7	**1.0**

Note: Results are averaged over six sessions per domain, covering underspecified requests, stable-preference requests, and preference-shift requests. Auto. Exec. denotes the percentage of execution dimensions completed at L_4. User Corr. denotes the percentage of sessions requiring user correction after autonomous execution.

emphasis on current-session coherence and more conservative automation-level adjustment. When $\bar{S}$ is large, increasing β and η reduces time cost by placing greater emphasis on current-session coherence and enabling faster progression toward higher automation levels. Table 6 analyzes the joint effect of τ, λ, and μ across the two task domains. Smaller values allow more task dimensions to proceed at L_4, reducing user action time. However, the increased agentic autonomy leads to more user correction actions. This trade-off is more pronounced in flight booking, where execution decisions such as booking submission are higher-stakes than meeting scheduling. Larger values make the system more conservative by requiring stronger stability before automation-level escalation. As a result, the correction rate decreases substantially, at the cost of increased interaction time.

D PROGRESS-REPORT GENERATION

EASE issues progress reports when the trigger condition in Section 3.4 is satisfied. These updates summarize autonomous execution progress together with the task context needed to interpret it.

Content selection and surface realization. For each reportable execution state d, EASE defines a status phrase and a corresponding set of condition dimensions. In Eq. 6, $g_{\text{phs}}(d)$ returns the status phrase, whereas $g_{\text{con}}(X^{(n)})$ verbalizes all finalized values for those dimensions in $X^{(n)}$. When available, a structured execution outcome returned by the Result Integrator, such as the number of matching options or an availability result, is incorporated into the same report. The selected content is then realized using a state-specific phrase pattern. Table 7 lists the field mappings and example reports for the evaluated domains.

Report consolidation and execution flow. Reports use the latest finalized values in the current session, with user corrections taking precedence over inferred or reused values. When multiple reports are triggered in the same interaction step, EASE orders them by workflow stage and consol-

Table 7: Content selection and surface forms for progress reports in the evaluated task domains. Bracketed fields are populated from finalized session values or structured execution outcomes.

Domain	Execution state	Selected content	Example report
Flight booking	Search initiated	Origin, destination, date, trip type, passenger count, cabin class, stop constraint, and search platform(s)	"I will search [platform(s)] for [trip type] [cabin class] flights for [passenger count] from [origin] to [destination] on [date] [stop-constraint phrase]."
Flight booking	Search completed	Route, date, and structured search outcome, such as matching options, price-check status, or recommendation status	"Search completed for [route] on [date]. [Result summary]."
Meeting scheduling	Candidate time slots found	Attendee(s), requested time window, duration, and feasible time slots	"I found [number] time slots that work for [attendee(s)] [time-window phrase]. [Candidate-slot summary]."
Meeting scheduling	Meeting room found	Finalized time, location preference, required capacity or facilities, and available room	"A suitable room is available at [time]: [room] in [location], with [relevant facilities]."

idates them into a single update, stating shared task context only once. Raw tool calls, intermediate parameter updates, and other low-level execution traces are omitted. If execution cannot continue without additional user input, EASE pauses and elicits the required input using the interaction form selected at the current automation level. Consequential actions such as booking submission and invitation sending proceed only after user confirmation at L_3.

E TASK DATABASE

We construct structured task databases for the two evaluated domains: flight booking and meeting scheduling. These databases provide controllable task states for evaluating human–agent communication, including preference elicitation, automation-level adaptation, intervention handling, and progress reporting. Together, they cover diverse scenarios, such as underspecified requests, repeated preferences, preference shifts, multi-option selection, and high-risk execution decisions. The flight-booking domain contains 100 records spanning 10 routes and 5 search platforms, while the meeting-scheduling domain includes 30 anonymized contacts and 8 room resources. Representative dimensions and records are reported below.

E.1 FLIGHT BOOKING DATABASE

The flight-booking task database is organized in a JSON-style format, where records describe flight options for a given route retrieved from different search platforms. As shown in Table 8, the database covers representative flight-booking dimensions, including route, airport, cabin, direct-only constraint, search platform, and price comparison. For connecting flights, segment-level fields further specify the transfer city, transfer airport, and the arrival/departure times used to derive the layover duration.

E.2 MEETING SCHEDULING DATABASE

The meeting-scheduling task database is organized as JSON-style contact and room-resource records. The database contains anonymized contacts, identity attributes, organizational roles, recent interaction frequency, usual meeting preferences, and location-related fields. This structure supports attendee identification, meeting-type selection, time-slot preference reuse, duration selection, mode and location selection, agenda confirmation, and invitation sending. As shown in Table 9, the listed values are representative examples. The full task database contains more records following the same format.

F USER SIMULATOR

We implement a constrained LLM-based user simulator that supports realistic and reproducible multi-session evaluation across diverse interaction settings. For each session n, the simulated user

Table 8: Representative dimensions and example values in flight booking. The example values show connecting-flight itineraries. The full task database contains more records following the same format.

Field	Example value
`from`, `to`	London → Sapporo
`from_airport`	London Gatwick Airport (LGW)
`to_airport`	New Chitose Airport (CTS)
`stops`	1
`airline`	China Eastern; Air China
`flight`	MU202 → MU279; CA852 → CA169
`date`	2026-05-18
`depart`, `arrival`	11:35 am → 1:08 pm
`segment_city1`	Shanghai; Beijing
`segment_airport1`	Shanghai Pudong International Airport (PVG); Beijing Capital International Airport (PEK)
`segment_arrival1`	PVG: 5:55 am; PEK: 5:45 am
`segment_depart2`	PVG: 8:00 am; PEK: 8:00 am
`layover_duration`	PVG: 2h05m; PEK: 2h15m
`cabin`	Economy; Business
`price`, `source`	Economy: £397 (Kayak), £380 (Skyscanner); Business: £850 (Google Flights)

Table 9: Representative dimensions and example values in meeting scheduling. Semicolons separate different contacts, while slashes separate multiple values for the same contact. `interaction_frequency` denotes the number of interactions in the past month.

Dimension	Example value
`name`	Alex A.; Alex B.
`short_name`	Alex
`title`	Professor; Department Administrator
`role`	Faculty; Administration
`department`	Department A
`email`	`contact_01@anon.edu`; `contact_02@anon.edu`
`office`	Office A; Office B
`relationship_strength`	0.92; 0.35
`interaction_frequency`	18; 4
`usual_meeting_types`	Research discussion / project meeting; administrative meeting / programme query
`usual_time_slots`	11:00 am / 1:00 pm; 10:00 am / 2:00 pm
`usual_duration`	60 minutes; 30 minutes
`usual_mode`	Online; in-person
`usual_location`	Teams; Office B / Meeting Room
`room_resource`	Meeting Room R1; Meeting Room R2
`keywords`	Research / project / supervision; administration / programme

follows a session-specific target-value vector

$$\mathbf{V}_n = \{v_{n,d} \mid d \in \mathcal{D}_n^{\mathrm{tar}}\}, \tag{10}$$

where $\mathcal{D}_n^{\mathrm{tar}}$ denotes the set of interaction dimensions for which the simulated user has an intended value in session n, and $v_{n,d}$ denotes the corresponding target value. The construction of $\mathcal{D}_n^{\mathrm{tar}}$ and its target values is detailed in Appendix G. Other execution dimensions are governed by the task state and authorization rules rather than by reusable preference values. At each interaction step, the simulator is prompted with the current target-value vector, the agent's latest message, the dialogue history, and the available interaction form. It is instructed to respond only with information relevant to the agent's current request and to remain consistent with the specified target values. Depending on the interaction form, the simulator expresses the intended value in natural language, selects

Table 10: Evaluation of the user simulators across the two task domains.

Method	Human-likeness	Coherence	Goal Consistency
Flight Booking			
Prompt-only LLM	2.03 (0.60)	2.58 (0.55)	2.34 (0.69)
Constrained LLM (Ours)	**2.14 (0.54)**	**2.79 (0.41)**	**2.82 (0.39)**
Meeting Scheduling			
Prompt-only LLM	2.14 (0.55)	2.64 (0.51)	2.42 (0.66)
Constrained LLM (Ours)	**2.25 (0.50)**	**2.83 (0.38)**	**2.86 (0.35)**

Note: Scores are assigned by an independent LLM judge on a 3-point scale and averaged over 50 sampled simulator responses per method and domain. The best score is shown in **bold**, standard deviations are in parentheses. *Human-likeness* measures whether the response resembles natural human communication, *Coherence* measures consistency with the dialogue history, and *Goal Consistency* measures alignment with the specified target values. Both simulators use the same LLM backbone, task instances, preference trajectories, and dialogue contexts.

the matching candidate, or confirms a consistent proposal and otherwise provides a correction. We compare the constrained simulator with a prompt-only LLM simulator under identical task and preference settings, with the results reported in Table 10.

G Preference and shift generation

To instantiate the target-value vector $\mathbf{V}_n$ introduced in Appendix F, we distinguish between task-specific requirements and reusable preference defaults. Let $\mathcal{G}_n \subseteq \mathcal{D}$ denote the dimensions for which the current task specifies a requirement, with $g_{n,d}$ denoting the corresponding value. Let $\mathcal{P} \subseteq \mathcal{D}$ denote the dimensions for which a reusable preference default is maintained, with $p_{n,d}$ denoting its value in session n. The target dimensions for session n are therefore given by

$$\mathcal{D}_n^{\mathrm{tar}} = \mathcal{G}_n \cup \mathcal{P}. \tag{11}$$

For each target dimension, the value used by the user simulator is resolved as

$$v_{n,d} = \begin{cases} g_{n,d}, & d \in \mathcal{G}_n, \\ p_{n,d}, & d \in \mathcal{P} \setminus \mathcal{G}_n. \end{cases} \tag{12}$$

For each target dimension, the simulator follows the current task requirement when one is specified and otherwise follows the corresponding reusable preference. This ensures consistent responses whenever the agent asks about or proposes a value for that dimension.

Whether a dimension uses $g_{n,d}$ or $p_{n,d}$ is determined separately for each session. For example, if the current request names an attendee, the simulator uses that attendee as $g_{n,d}$. If the attendee is omitted, it may instead use the recurring attendee stored in $p_{n,d}$. The same rule applies to dimensions such as departure airport, meeting time, duration, and location. An explicit task requirement overrides the corresponding preference only for the current session and does not change $p_{n,d}$. We treat a change as a preference shift only when $p_{n,d}$ itself is updated and carried forward to subsequent sessions.

Initial preference generation. The reusable preference dimensions instantiated for a trajectory are selected from the task-relevant dimensions supported by the corresponding database. In flight booking, these may include route, departure and arrival airports, cabin class, stop constraint, passenger count, and search platform(s). In meeting scheduling, they may include attendee, meeting type, preferred time, duration, mode, location, and agenda preference. The exact subset depends on the task template and the cross-session trajectory being evaluated.

For each reusable dimension $d \in \mathcal{P}$, the initial preference $p_{1,d}$ is sampled from the feasible values in its domain $\mathcal{A}_d$. When multiple dimensions are subject to a shared constraint, their values are sampled jointly from valid combinations. For example, an online meeting is paired with an online platform, whereas an in-person meeting is paired with an office or meeting room.

For the current task, an explicitly specified requirement overrides any reusable preference on the same dimension according to Eq. equation 12. This rule determines only the simulator's target value. Each evaluated method independently decides how to incorporate the interaction into its history or user profile. Unless the user indicates that the new value should carry over to future sessions, the simulator leaves the corresponding reusable preference $p_{n,d}$ unchanged.

Stable preferences. Let $\mathcal{N}_{\text{stable}}$ denote the sessions in the stable-preference phase. Throughout this phase, each reusable preference retains its initial value:

$$p_{n,d} = p_{1,d}, \qquad d \in \mathcal{P}, \quad n \in \mathcal{N}_{\text{stable}}. \tag{13}$$

Task-specific requirements may differ across sessions. When a requirement is explicitly specified, it overrides the corresponding preference for that session according to Eq. equation 12, without changing the stored value $p_{n,d}$. We therefore treat such cases as session-specific requirements rather than preference shifts. This setting measures whether a method can learn recurring preferences from earlier interactions and reuse them with less user input in later sessions.

Changing preferences. Let $\mathcal{N}_{\text{shift}}$ denote the sessions containing one or more preference shifts. For each $n \in \mathcal{N}_{\text{shift}}$, let $\mathcal{S}_n \subseteq \mathcal{P}$ denote the non-empty set of affected preference dimensions. The size of $\mathcal{S}_n$ may vary across sessions, allowing a shift to affect either one or multiple dimensions. Candidate shift dimensions are those relevant to the current task, unspecified in the current request, and associated with at least one feasible alternative. One or more candidate dimensions are then randomly chosen to form $\mathcal{S}_n$.

Let $\mathcal{A}_d^{(n)} \subseteq \mathcal{A}_d$ denote the values that are feasible under the current task and tool state. For each affected dimension, the reusable preference is sampled after excluding its preceding value:

$$p_{n,d} \sim \text{Uniform}\left(\mathcal{A}_d^{(n)} \setminus \{p_{n-1,d}\}\right), \qquad d \in \mathcal{S}_n. \tag{14}$$

When affected dimensions are subject to shared constraints, their values are sampled jointly from feasible combinations. Reusable preferences outside the affected set retain their preceding values:

$$p_{n,d} = p_{n-1,d}, \qquad d \in \mathcal{P} \setminus \mathcal{S}_n. \tag{15}$$

The resulting values remain in effect in subsequent sessions until the corresponding dimensions change again. A changing-preference session may therefore involve one or more persistent preference updates, while preferences outside $\mathcal{S}_n$ remain unchanged.

Shift exposure and experimental control. The evaluated method does not receive explicit labels indicating which preferences differ from previously established values. Instead, such differences become observable through human–agent interaction when the corresponding dimensions become relevant. If the agent asks about an affected dimension, the simulator provides its current target value. If the agent proposes or reuses a value that is inconsistent with the current target, the simulator rejects it and provides a concise correction. If the agent proceeds autonomously, the simulator intervenes when the inconsistent value first appears in a user-visible proposal or progress report, before any booking or invitation is finalized. Values on unaffected dimensions are not volunteered unless they are explicitly queried or included in an agent proposal.

All methods use the same task instances, preference trajectories, and random seeds for user simulation. For each changing-preference session, the shift record is defined as

$$\mathcal{R}_n^{\text{shift}} = \left\{ \left(d, p_{n-1,d}, p_{n,d}\right) \mid d \in \mathcal{S}_n \right\}, \tag{16}$$

which records the affected dimensions together with their preceding and current values. These records provide the reference values used to measure shift-handling cost and automation-level adaptation.

H DIALOGUE EXAMPLES

We present condensed dialogue traces in Tables 11–14 to show how EASE adapts human–agent communication under both stable and evolving user preferences. Under stable preferences, repeated task values are gradually converted from free-text specification into lower-effort selections,

Table 11: Flight-booking dialogue under stable preferences. Quoted text denotes free-text user input, while bracketed entries denote lightweight user actions such as option clicks or confirmation buttons. The trace reports representative interaction stages rather than raw terminal logs, showing how EASE converts repeated preferences into lower-effort communication while retaining explicit user confirmation for high-stakes decisions.

Interaction stage	Session 1	Session 3	Session 6
User request	"Find cheap flights from [origin city] to [destination city] this week."	"Find flights this week."	"Find flights this week."
Condition and preference resolution			
Route	L_1 specification *typed:* [origin] → [destination]	L_2 selection *[Select: usual route]*	L_3 confirmation *[Confirm usual route]*
Departure airport(s)	L_1 specification *typed:* [departure airport(s)]	L_2 selection *[Select: airport option(s)]*	L_3 confirmation *[Confirm usual departure airport(s)]*
Arrival airport(s)	L_1 specification *typed:* [arrival airport(s)]	L_2 selection *[Select: airport option(s)]*	L_3 confirmation *[Confirm usual arrival airport(s)]*
Passenger count	L_1 specification "[number] adult(s)"	L_2 selection *[Select: usual passenger count]*	L_3 confirmation *[Confirm usual passenger count]*
Cabin class	L_2 selection *[Select: cabin class]*	L_3 confirmation *[Confirm usual cabin class]*	L_4 full automation *[Auto-fill usual cabin class]*
Direct-only constraint	L_2 selection *[Select: direct/connection preference]*	L_3 confirmation *[Confirm usual flight constraint]*	L_4 full automation *[Auto-fill usual flight constraint]*
Search platforms	L_1 specification *typed/blank:* [platform set]	L_2 selection *[Select: platform option(s)]*	L_3 confirmation *[Confirm usual platform set]*
Execution, key progress reporting, and high-stakes confirmation			
Search / price-check actions	L_3 confirmation *[Confirm search]* *[Confirm price check]*	L_4 full automation *[Auto-start search]* *[Auto-verify price]*	L_4 full automation *[Auto-start search]* *[Auto-verify price]*
Key progress report	searching selected platform(s); results found	searching usual platform(s); price checked	search completed; recommended option ready
Flight choice	L_2 selection *[Select: recommended option]*	L_2 selection *[Select: recommended option]*	L_2 selection *[Select: recommended option]*
Booking submission	L_3 confirmation *[Confirm booking]*	L_3 confirmation *[Confirm booking]*	L_3 confirmation *[Confirm booking]*

Note: Bracketed actions are selections or confirmations, not typed user responses. Airport and platform dimensions may contain one or more selected values. The key progress report summarizes decision-relevant progress while omitting low-level tool-use traces.

confirmations, or autonomous execution. Under evolving preferences, user interventions trigger automation-level downgrades only on the affected dimensions, allowing the updated preference to be incorporated without resetting the entire workflow. Low-level tool-use traces are omitted, while only key progress reports and decision-relevant outcomes are retained.

I Additional experimental results

Tables 15 and 16 report user input tokens across interaction rounds for the flight-booking and meeting-scheduling agents, respectively. All experiments were run on a workstation equipped with an Intel Core i7-14700 CPU, 64 GB DDR5 ECC memory, an NVIDIA RTX 4000 Ada GPU with 20 GB memory, and a 2 TB PCIe M.2 SSD. Across both task domains, EASE consistently achieves the lowest input cost under various settings. Under stable preferences, EASE shows the largest reduction from the first to the sixth round, indicating that repeated user regularities are effectively converted into lower-cost interaction forms. Static-Profile is usually the closest baseline in the stable setting because it can reuse historical preferences, but it remains worse than EASE since it lacks online automation-level adjustment. Under changing preferences, EASE maintains substantially lower input tokens than CIPHER and Static-Profile, showing that latent or static preference reuse alone is insufficient when user requirements evolve. Compared with DTAC, EASE further reduces input burden by jointly adapting when user input is needed and how it is elicited.

Table 12: Flight-booking dialogue under changing preferences. Quoted text denotes user-typed free-text intervention, while bracketed entries denote lightweight user actions such as option clicks or confirmation buttons. The table highlights how EASE downgrades automation on affected dimensions after a preference shift, incorporates the updated preference through low-cost interaction, and preserves explicit confirmation for high-stakes task decisions.

Interaction stage	Before preference shift	User intervention	After adaptation
User request	"Find flights this week." Usual preferences inferred	"Actually, [updated preference]."	Updated preference stored in profile
Route	L_3 confirmation *[Confirm usual route]*	No change	L_3 confirmation *[Confirm usual route]*
Departure / arrival airport(s)	L_3 confirmation *[Confirm usual airport(s)]*	No change	L_3 confirmation *[Confirm usual airport(s)]*
Direct-only constraint	L_3 confirmation *[Confirm usual constraint]*	L_0 user override "Actually, [new constraint]."	L_2 selection *[Select: updated constraint]*
Search platforms	L_3 confirmation *[Confirm usual platform set]*	L_0 user override "Skip [platform]."	L_2 selection *[Select: updated option(s)]*
Search / price-check actions	L_4 full automation *[Auto-start search]* *[Auto-verify price]*	Search criteria changed upstream	L_3 confirmation *[Confirm updated criteria]*
Key progress report	search completed; recommended option ready	Preference shift detected	criteria updated; new search completed
Flight choice	L_2 selection *[Select: recommended option]*	No direct change	L_2 selection *[Select: updated option]*
Booking submission	L_3 confirmation *[Confirm booking]*	High-stakes decision	L_3 confirmation *[Confirm booking]*

Note: The key progress report is an execution output, not an interaction dimension.

Table 13: Meeting-scheduling dialogue under stable preferences. Quoted text denotes free-text user input, while bracketed entries denote lightweight user actions such as option clicks or confirmation buttons. The trace reports representative interaction stages rather than raw terminal logs, showing how EASE converts repeated meeting preferences into communication-efficient actions while retaining explicit user confirmation for high-stakes invitation sending.

Interaction stage	Session 1	Session 3	Session 6
User request	"Arrange me a meeting with [contact]."	"Arrange the usual meeting with [contact]."	"Arrange the usual meeting."
Condition and preference resolution			
Contact identity	L_3 confirmation *[Confirm inferred contact]*	L_3 confirmation *[Confirm usual contact]*	L_4 full automation *[Auto-fill usual contact]*
Meeting type	L_2 selection *[Select: meeting type]*	L_3 confirmation *[Confirm usual meeting type]*	L_4 full automation *[Auto-fill usual type]*
Time slot	L_1 specification "[date] at [time]"	L_2 selection *[Select: usual time option]*	L_3 confirmation *[Confirm usual time slot]*
Duration	L_1 specification "[duration]"	L_3 confirmation *[Confirm usual duration]*	L_4 full automation *[Auto-fill usual duration]*
Mode	L_2 selection *[Select: meeting mode]*	L_3 confirmation *[Confirm usual mode]*	L_4 full automation *[Auto-fill usual mode]*
Location	L_2 selection *[Select: location/link option]*	L_3 confirmation *[Confirm usual location]*	L_4 full automation *[Auto-fill usual location]*
Agenda	L_3 confirmation *[Confirm agenda setting]*	L_3 confirmation *[Confirm usual agenda setting]*	L_4 full automation *[Auto-fill usual agenda setting]*
Execution, key progress reporting, and high-stakes confirmation			
Availability / calendar preparation	L_3 confirmation *[Confirm availability check]*	L_4 full automation *[Auto-check availability]* *[Auto-prepare invite]*	L_4 full automation *[Auto-check availability]* *[Auto-prepare invite]*
Key progress report	availability checked; invitation draft ready	usual slot checked; invitation draft ready	calendar event prepared; invitation ready
Invitation sending	L_3 confirmation *[Confirm sending invitation]*	L_3 confirmation *[Confirm sending invitation]*	L_3 confirmation *[Confirm sending invitation]*

Note: Bracketed actions are selections, confirmations, or agentic auto-fill actions, not typed user responses. Attendee, location, and agenda-related dimensions may be inferred from stable meeting preferences or confirmed contextual information. The key progress report summarizes decision-relevant calendar progress while omitting low-level scheduling and tool-use traces.

J STRESS TESTING ON PUBLIC TOOL–AGENT–USER BENCHMARKS

We evaluate EASE on public tool–agent–user benchmarks to examine whether its communication-efficiency gains preserve reliable task execution under challenging conditions. We focus on cases

Table 14: Meeting-scheduling dialogue under preference evolution. Quoted utterances indicate user-initiated free-text corrections, whereas bracketed entries indicate lightweight user actions such as selections or confirmations. This example shows how EASE responds to changing meeting preferences by dynamically adjusting automation levels and re-validating the updated meeting details.

Interaction stage	Before preference shift	User intervention	After adaptation
User request	"Arrange the usual meeting."	"Actually, [updated meeting preference]."	Updated preference stored in profile
Contact identity	L_3 confirmation *[Confirm usual contact]*	No change	L_3 confirmation *[Confirm usual contact]*
Meeting type	L_4 full automation *[Auto-fill usual type]*	L_0 user override "Make it [new type]."	L_2 selection *[Select: updated type]*
Time slot	L_3 confirmation *[Confirm usual time slot]*	L_0 user override "Move it to [new time]."	L_2 selection *[Select: updated time option]*
Duration	L_4 full automation *[Auto-fill usual duration]*	L_0 user override "Make it [new duration]."	L_2 selection *[Select: updated duration]*
Mode	L_4 full automation *[Auto-fill usual mode]*	L_0 user override "Make it [new mode]."	L_2 selection *[Select: updated mode]*
Location	L_4 full automation *[Auto-fill usual location]*	Mode/location changed upstream	L_2 selection *[Select: updated location]*
Agenda	L_4 full automation *[Auto-fill usual agenda setting]*	L_0 user override "No agenda this time."	L_3 confirmation *[Confirm updated agenda setting]*
Availability / calendar preparation	L_4 full automation *[Auto-check availability]* *[Auto-prepare invite]*	Meeting criteria changed upstream	L_3 confirmation *[Confirm updated meeting details]*
Key progress report	calendar event prepared; invitation ready	Preference shift detected	criteria updated; availability rechecked
Invitation sending	L_3 confirmation *[Confirm sending invitation]*	High-stakes decision	L_3 confirmation *[Confirm sending invitation]*

Note: The key progress report is an execution output, not an interaction dimension.

Table 15: User input tokens to the flight agent across interaction rounds. Stable preferences are evaluated in $S1 - S6$, while preferences change in $S7 - S12$. Avg. denotes the average input tokens within each setting, and ΔS denotes the change in input tokens from S_1 to S_6 under stable preferences. Best/second-best results are marked in bold/underlined. Improv. reports the relative reduction of EASE over the second-best method where applicable.

Methods	Stable Preference								Changing Preference						
	S1	S2	S3	S4	S5	S6	Avg.	ΔS	S7	S8	S9	S10	S11	S12	Avg.
Inf.-Drv.	27	27	27	27	27	27	27.00	0	35	36	35	34	36	35	35.17
Sgl.-Trn.	**21**	21	21	21	21	21	21.00	0	29	30	28	30	29	28	29.00
Sta.-Prof.	22	20	17	15	13	12	16.50	-10	24	26	25	26	25	24	25.00
DTAC	25	25	22	22	20	20	22.33	-5	25	27	26	27	26	25	26.00
CIPHER	26	22	21	19	15	15	19.67	**-11**	26	28	27	27	26	26	26.67
EASE	**21**	**17**	**14**	**12**	**10**	**10**	**14.00**	**-11**	**18**	**21**	**20**	**23**	**21**	**20**	**20.33**
Improv.	–	15.00%	17.65%	20.00%	23.08%	16.67%	15.15%	–	25.00%	19.23%	20.00%	11.54%	16.00%	16.67%	18.68%

where user requests are underspecified, current preferences conflict with the learned user profile, or available flights, attendees, or time slots change during task execution.

Benchmarks. For flight booking, we use 20 tasks from the airline domain of τ-bench (Yao et al., 2024), which evaluates an agent through interaction with a simulated user, domain tools, and a stateful reservation database. Following its evaluation protocol, task success is determined by comparing the final database state with the annotated goal state. For meeting scheduling, we construct a 20-task subset of Asper-Bench (Coca et al., 2025), covering attendee resolution, availability checking, constrained scheduling, event creation or rescheduling, room and modality selection, and invitation preparation. We augment each task with cross-session interaction histories and expose its user-facing decisions through the interaction forms supported by EASE.

Multi-session evaluation protocol. We extend each benchmark task into a four-session sequence. The first three sessions provide repeated interactions from which history-aware methods can learn user preferences, while the fourth session is used for evaluation. Before evaluation, all bookings, calendar events, and resource availability are reset to their original benchmark states.

For each domain, we evaluate 20 tasks. 4 tasks preserve the preferences established in the preceding sessions. The remaining 16 tasks are evenly divided among 4 stress conditions: (i) missing or

Table 16: User input tokens to the meeting agent across interaction rounds. Stable and changing preferences are each evaluated over six rounds. Avg. denotes the average input tokens within each setting, and ΔS denotes the token reduction from $S1$ to $S6$ under stable preferences. Best/second-best results are marked in bold/underlined. Improv. reports the relative reduction of EASE over the second-best method where applicable.

Methods	Stable Preference								Changing Preference						
	S1	S2	S3	S4	S5	S6	Avg.	ΔS	S1	S2	S3	S4	S5	S6	Avg.
Inf.-Drv.	18	18	18	18	18	18	18.00	0	24	25	24	25	24	24	24.33
Sgl.-Trn.	**14**	14	14	14	14	14	14.00	0	20	21	20	21	20	20	20.33
Sta.-Prof.	15	14	13	12	11	11	12.67	-4	18	20	19	20	19	18	19.00
DTAC	16	16	15	15	14	14	15.00	-2	19	21	20	21	20	19	20.00
CIPHER	17	15	14	13	12	12	13.83	-5	20	22	21	21	20	20	20.67
EASE	**14**	**12**	**10**	**9**	**8**	**8**	**10.17**	**-6**	**15**	**18**	**17**	**18**	**17**	**16**	**16.83**
Improv.	–	14.29%	23.08%	25.00%	27.27%	27.27%	19.73%	–	16.67%	10.00%	10.53%	10.00%	10.53%	11.11%	11.42%

Table 17: Stress-injection protocol. Each evaluation task contains at most one injected perturbation.

Stress condition	**Injection point**	**Injected perturbation**	**Flight-booking instantiation**	**Meeting-scheduling instantiation**	**Expected agent response**
Stable control	No injection	No perturbation is introduced. Session-specific task details may change in S_{inj} without contradicting the learned profile.	The destination or travel date changes, while established preferences such as cabin class, departure airport, direct-only constraint, and search platform remain unchanged.	The attendee, subject, or date changes, while established preferences such as duration, modality, and location remain unchanged.	Reuse stable preferences where appropriate while satisfying the current request, without unnecessary user elicitation.
Missing or ambiguous information	Initial user request in S_{inj}	One decision-relevant value is omitted or made ambiguous. The intended value is retained in the hidden user goal and can be resolved through targeted interaction.	The user specifies "from London" while multiple departure airports lead to different feasible itineraries, or omits whether a connection is acceptable when this changes the selected flight.	The user refers to "Alex" when multiple contacts share that name, or requests a meeting on "Thursday afternoon" when several available slots, such as 1:00–2:00 and 3:00–4:00, satisfy the request.	Infer the value when the observable context uniquely determines it; otherwise, request targeted specification or present a small set of candidate options.
Current–history conflict	Initial request or user correction in S_{inj}	One previously stable preference is changed. All unrelated historical preferences remain valid.	A user who previously requested direct flights states that connections are acceptable for the current trip, or changes the usual cabin, airport, or search platform.	A user who usually schedules online 60-minute meetings requests an in-person or 30-minute meeting for the current task.	Use the user's current preference when it conflicts with the historical profile, update the affected dimension, and continue reusing unaffected stable preferences.
Tool-state change	After the first successful search or availability check, but before the final state-changing tool call	A previously feasible option is made unavailable or changed in a way that affects the current plan. At least one valid alternative remains when the task is designed to be solvable.	The selected flight becomes unavailable, its fare changes, or the requested cabin sells out after the initial search.	The selected room becomes occupied, an attendee becomes unavailable, or the chosen time slot is removed after the initial availability check.	Refresh the relevant tool state, discard the stale plan, and identify a valid alternative. Re-engage the user when the alternative changes a user-facing decision such as price, itinerary, attendee, time, or room.
Confirmation-critical action	Before the final state-changing tool call	User confirmation is required after the final action details have been presented. If any decision-relevant detail changes afterward, confirmation must be requested again.	Before booking, the agent must present the final route, airports, date, passenger count, itinerary, cabin class, number of stops, search platform, and total price.	Before sending an invitation, the agent must present the attendee, subject, date, time, duration, modality, location, room, and agenda.	Execute the final booking or scheduling action only after receiving user confirmation of the final details. If any decision-relevant detail changes, request confirmation again.

Note: S_{inj} denotes the session in which the perturbation is introduced. Tool-state changes are triggered by predefined execution events rather than fixed dialogue turns, ensuring that all methods encounter the perturbation at the same stage of task execution. The task environment is reset before each task–seed run.

ambiguous task information, (ii) a current request that conflicts with a previously stable preference, (iii) a suitable flight, attendee, room, or time slot becoming unavailable, and (iv) a consequential action that requires user confirmation, such as submitting a booking or sending a meeting invitation. Table 17 summarizes the injection point, task-specific examples, and expected agent response for each stress condition in S_{inj}. Each evaluation task is run three times with different user-simulator seeds to account for variations in user responses. All methods are evaluated under the same tasks, tools, and interaction affordances.

Metrics. We report *Task Success* and pass^3 as the main performance metrics. A task run is successful if all user-specified constraints are satisfied and no unintended booking or calendar change is made. For flight booking, we check the route, date, airports, passenger count, cabin class, flight constraints, selected itinerary, and final reservation. For meeting scheduling, we check the attendees,

Table 18: Representative error cases in the benchmark-derived stress tests.

(a) Task-completion error examples.

ID	User request and injected context	Agent action
1	**Flight; current–history conflict.** The user historically requested direct flights but states in S_4: "Connections are acceptable this time if they reduce the price."	**Agent:** "I searched for direct flights based on your usual preference, but no itinerary meets your budget."
Error cause: *Stale-preference reuse.* The agent gives the historical direct-only constraint precedence over the user's current preference. It should update the affected preference and search both direct and one-stop itineraries.		
2	**Meeting; tool-state change.** A room identified during the initial availability check becomes occupied before event creation.	**Agent:** "The meeting is ready for Thursday at 2 PM in Room A." The agent terminates without refreshing room availability, and no valid event is created.
Error cause: *Stale-state execution.* The response relies on an observation invalidated by the injected state change. The agent should refresh availability, identify a feasible alternative, and complete the requested event.		
3	**Flight; missing or ambiguous information.** The user requests a flight "from London," but does not specify the departure airport. Both Heathrow and Gatwick offer feasible but different itineraries, so the intended airport cannot be determined from the current request.	The agent assumes Heathrow, searches only flights departing from Heathrow, and returns an itinerary that otherwise satisfies the request.
Error cause: *Premature commitment under ambiguity.* The agent selects a departure airport before the ambiguity is resolved, although multiple feasible airports remain consistent with the current request.		

(b) Unintended-action error examples.

ID	User request and injected context	Agent action
1	**Flight; tool-state change.** The user selects Flight A, but it becomes unavailable before booking. Flight B departs later and costs more.	Without informing the user, the agent calls `book_flight(Flight B, ...)` and reports: "Flight B has been booked."
Error cause: *Unapproved itinerary substitution.* Although Flight B remains feasible, its departure time and price differ from those of the option selected by the user. The agent should report that Flight A is no longer available and obtain user confirmation of the replacement itinerary before booking.		
2	**Meeting; entity ambiguity.** Two contacts named Alex match the information provided in the current request.	The agent selects the contact most frequently chosen in prior sessions and calls `send_invitation(Alex A, ...)`, although the user intended Alex B.
Error cause: *Entity-binding failure.* The ambiguous reference to "Alex" is bound to the wrong contact, and the error propagates to invitation sending. Interaction history may rank the candidates but cannot uniquely identify the intended attendee. The agent should present the matching contacts for user selection before sending the invitation.		
3	**Meeting; confirmation-critical action.** The agent has assembled and presented the proposed event, but the user has not confirmed the final time and room.	The agent calls `create_event(...)` and `send_invitation(...)` after presenting the event summary.
Error cause: *Confirmation-timing error.* The agent executes the scheduling actions before receiving user confirmation of the final event details. Presenting an event summary does not itself constitute user confirmation. The agent should wait for confirmation before creating the event and sending the invitation.		

time, duration, mode, location, availability constraints, and final calendar event. Following τ-bench, pass^3 is the percentage of tasks successfully completed in all 3 independent runs with different user-simulator seeds. It captures the reliability of agents in interacting with humans and tools in dynamic and realistic settings.

For the stress-test tasks, we report 3 error types. A *task-completion error* occurs when the final result violates a user-specified constraint. An *unintended-action error* occurs when tool execution causes an unintended change, such as creating a duplicate reservation, modifying an unrelated calendar event, or inviting the wrong attendee. A *handback-control error* occurs when the agent fails to

Table 18: Representative error cases in the benchmark-derived stress tests (continued).

(c) Handback-control error examples.

ID	User request and injected context	Agent action
1	**Flight; confirmation invalidated by a fare change.** The user confirms Flight A at £380, but its fare changes to £470 before booking.	The agent reuses the earlier confirmation and books Flight A at the updated price without asking the user again.
Error cause: *Reliance on stale confirmation.* The earlier confirmation applies to Flight A at £380 and does not cover the updated fare of £470. The agent should report the fare change and request confirmation again before booking.		
2	**Meeting; unresolved attendee ambiguity.** The request says "Schedule a meeting with Alex," and two contacts match the available information.	The agent autonomously selects the contact who appears most frequently in the historical profile.
Error cause: *Missed handback despite unresolved ambiguity.* The agent commits to one contact before the intended attendee has been uniquely identified. The historical profile may rank the candidates but cannot resolve the remaining ambiguity. The agent should present the matching contacts for user selection.		
3	**Meeting; uniquely identifiable event.** The user asks to move tomorrow's project meeting to 3 PM, and only one calendar event matches the request.	**Agent:** "Which project meeting do you mean? Please provide the attendees, original time, and subject."
Error cause: *Unnecessary handback.* The current request and calendar state uniquely identify the target event. The agent should resolve the event directly and request confirmation only before applying the rescheduling change.		

request user input when it is needed, or requests additional input when the available information is already sufficient. A failed run may belong to more than one error category. Table 18 provides representative cases illustrating how these errors arise under the injected stress conditions.

EASE remains reliable as tool states change and user preferences evolve. As shown in Table 19, in flight booking, missing or conflicting constraints and changes in availability can directly alter the feasible itinerary. When such unresolved information could change the booking outcome, DTAC is more likely to seek clarification before proceeding, making it the strongest baseline in this domain. In meeting scheduling, preferences over attendees, duration, mode, and location recur more often across sessions, allowing CIPHER to effectively reuse patterns inferred from prior user edits. Nevertheless, EASE improves task success by 6.7 and 5.0 percentage points, and pass^3 by 15.0 and 10.0 percentage points, over the respective strongest baselines in flight booking and meeting scheduling. These gains arise from EASE's dimension-wise calibration of autonomy that prevents stale or uncertain preferences from propagating into tool actions.

EASE suppresses error propagation and unnecessary handbacks under stress conditions. The error breakdown in Table 19 shows distinct failure patterns across the three categories. Even a single violation of a booking or meeting constraint can produce a task-completion error, making it more frequent than an unintended-action error, which occurs only when an incorrect decision propagates further to a tool call. Handback-control errors, by contrast, are more common when a method misjudges whether the available information is sufficient to proceed or additional user input is needed. Accordingly, DTAC reduces task-completion and unintended-action errors at the cost of more handbacks, whereas CIPHER avoids some handbacks through preference learning but becomes less reliable when the inferred preferences become stale. EASE lowers all three error rates by more than 50% relative to the strongest baselines. Its adaptive interaction policy selectively solicits user input at decision points that can alter the outcome, intercepting uncertain inferences before consequential tool execution without imposing repeated handbacks.

K Robustness to user-input errors, profile errors, and over-automation

Following the perturbation pipeline in DRIFT-BENCH (Bao et al., 2026), our robustness evaluation consists of three phases: (1) *target selection*, where we select one task-relevant dimension whose

Table 19: Performance on public tool–agent–user benchmarks. TS denotes Task Success. All values are percentages. Error categories are non-exclusive. Bold/underlined: best/second-best results.

Method	Overall Benchmark Results				Error Rates on Stress-Test Tasks		
	Flight Booking		Meeting Scheduling		Task-Comp. Err. ↓	Unintended-Action Err. ↓	Handback-Control Err. ↓
	TS ↑	pass^3 ↑	TS ↑	pass^3 ↑			
Single-Turn	76.7	50.0	73.3	45.0	18.8	10.4	21.9
Static-Profile	78.3	55.0	76.7	50.0	16.7	12.5	19.8
Inference-Driven	81.7	60.0	80.0	55.0	15.6	8.3	17.7
DTAC	<u>88.3</u>	<u>70.0</u>	83.3	60.0	<u>11.5</u>	<u>6.3</u>	12.5
CIPHER	86.7	65.0	<u>88.3</u>	<u>70.0</u>	12.5	7.3	<u>10.4</u>
EASE	**95.0**	**85.0**	**93.3**	**80.0**	**5.2**	**3.1**	**4.2**
Improv.	+6.7 pp	+15.0 pp	+5.0 pp	+10.0 pp	54.8%	50.8%	59.6%

Note: The evaluated tasks are derived from the original benchmarks but are augmented with cross-session interaction histories, controlled tool-state perturbations, and additional user-facing decision points. Therefore, the reported results are not directly comparable with those obtained under the original benchmark protocols.

value affects a downstream agent action and retain its correct value in the hidden user goal. (2) *perturbation generation*, where we construct two types of errors—a user-response error that replaces the intended response with a feasible but incorrect alternative, and a preference-profile error that replaces one retained preference with a feasible but incorrect value. (3) *perturbation injection*, where the user-response error is introduced at the corresponding elicitation turn, whereas the preference-profile error is inserted after the interaction history has been established. When the perturbed value appears in the agent output, the simulated user corrects it to the intended value. All non-target task values, preferences, and tool states remain unchanged. To examine the potential harm of over-automation, each perturbation is applied to a dimension that affects a subsequent state-changing action, and we measure whether the error is exposed and corrected before booking submission or invitation sending.

To isolate the effect of risk-aware automation control, we evaluate an EASE w/o Risk Control variant in which consequential action dimensions, such as booking submission and invitation sending, follow the same automation-level update as other task dimensions. Apart from task success, we report three robustness metrics: *Correction Rate*, the percentage of injected errors corrected before the corresponding state-changing action; *Consequential-Action Error Rate*, the percentage of runs in which booking submission or invitation sending is executed with an incorrect value; and *Correction Turns*, the number of interaction turns required to restore a task state consistent with the hidden user goal, averaged over successfully corrected runs.

EASE enables efficient error recovery while limiting propagation to consequential actions. As shown in Table 20, across both domains and error types, EASE outperforms the baselines by over 6.7 percentage points in task success and correction rate, while reducing consequential action errors and correction turns by at least 48.5% and 16.7%, respectively. For user-input errors, most methods perform slightly better in meeting scheduling than in flight booking, with higher correction rates and lower consequential-action error rates. A wrong meeting time, duration, or location can often be revised through another availability check, whereas an incorrect airport, cabin, or itinerary constraint can invalidate the retrieved flight options and propagate to the final reservation, potentially resulting in the wrong flight being booked. Preference-profile errors cause larger task-success drops than user-input errors, particularly for Static-Profile and CIPHER. A user-input error is confined to the current dialogue context and becomes observable once the incorrect value is incorporated into a subsequent agent output. By contrast, a corrupted profile value persists in the cross-session state and may be retrieved as context for multiple downstream decisions. Single-Turn is therefore most sensitive to user-input errors because it cannot compare the response with prior behaviour, whereas Static-Profile and CIPHER are more vulnerable to profile errors because their decisions are conditioned on retained or inferred historical preferences. Although DTAC prevents many errors through additional user elicitation, this robustness comes at the cost of substantially more correction turns. By updating only the affected dimension while retaining the remaining resolved task state, EASE achieves high correction rates with few correction turns. Its risk-aware automation-level control further prevents residual errors from reaching consequential actions, ensuring that user-input and profile errors do not result in unintended tool execution.

Table 20: Robustness to user-input errors, preference-profile errors, and over-automation perturbations. TS denotes Task Success, with the absolute change from the corresponding unperturbed task shown in parentheses. CR, CAE, and CT denote Correction Rate, Consequential-Action Error Rate, and Correction Turns, respectively. Bold/underlined: best/second-best results.

Method	Flight Booking				Meeting Scheduling			
	TS ↑	CR ↑	CAE ↓	CT ↓	TS ↑	CR ↑	CAE ↓	CT ↓
(a) User-input errors								
Inference-Driven	70.0 (−11.7)	75.0	16.7	2.2	71.7 (−8.3)	78.3	13.3	2.0
Single-Turn	60.0 (−16.7)	63.3	26.7	2.5	61.7 (−11.6)	68.3	21.7	2.3
Static-Profile	70.0 (−8.3)	78.3	13.3	1.9	71.7 (−5.0)	81.7	10.0	1.8
DTAC	83.3 (−5.0)	90.0	<u>5.0</u>	1.5	80.0 (−3.3)	91.7	<u>3.3</u>	1.4
CIPHER	81.7 (−5.0)	86.7	8.3	1.7	85.0 (−3.3)	90.0	6.7	1.6
EASE w/o Risk Control	<u>88.3</u> (−6.7)	<u>93.3</u>	6.7	<u>1.4</u>	<u>90.0</u> (−3.3)	<u>95.0</u>	5.0	<u>1.3</u>
EASE	**93.3** (**−1.7**)	**98.3**	**1.7**	**1.1**	**91.7** (**−1.6**)	**98.3**	**1.7**	**1.1**
(b) Preference-profile errors								
Static-Profile	58.3 (−20.0)	61.7	28.3	2.7	61.7 (−15.0)	66.7	23.3	2.5
CIPHER	73.3 (−13.4)	76.7	16.7	2.2	80.0 (−8.3)	81.7	13.3	2.0
EASE w/o Risk Control	<u>85.0</u> (−10.0)	<u>90.0</u>	<u>8.3</u>	<u>1.6</u>	<u>86.7</u> (−6.6)	<u>93.3</u>	<u>6.7</u>	<u>1.5</u>
EASE	**91.7** (**−3.3**)	**96.7**	**1.7**	**1.3**	**88.3** (**−5.0**)	**95.0**	**3.3**	**1.4**
(c) Over-automation perturbations								
Inference-Driven	65.0 (−16.7)	68.3	21.7	2.4	68.3 (−11.7)	73.3	16.7	2.2
Single-Turn	53.3 (−23.4)	56.7	33.3	2.8	56.7 (−16.6)	61.7	26.7	2.6
Static-Profile	58.3 (−20.0)	63.3	28.3	2.6	61.7 (−15.0)	68.3	23.3	2.4
DTAC	78.3 (−10.0)	83.3	10.0	1.8	76.7 (−6.6)	85.0	8.3	1.7
CIPHER	73.3 (−13.4)	78.3	15.0	2.0	80.0 (−8.3)	83.3	11.7	1.9
EASE w/o Risk Control	<u>83.3</u> (−11.7)	<u>88.3</u>	<u>8.3</u>	<u>1.6</u>	<u>85.0</u> (−8.3)	<u>90.0</u>	<u>6.7</u>	<u>1.5</u>
EASE	**90.0** (**−5.0**)	**93.3**	**5.0**	**1.5**	**91.7** (**−1.6**)	**96.7**	**1.7**	**1.1**

Note: Panel (b) includes methods that maintain and reuse cross-session preference information. Methods without a reusable preference profile are not affected by the profile perturbation and are therefore omitted. In panel (c), the first interaction that would otherwise re-engage the user on an affected dimension is suppressed, forcing the method to proceed autonomously with the retained value. Subsequent progress reporting, user intervention, and consequential-action safeguards remain available.

EASE maintains reliable execution under over-automation through timely correction and risk-aware safeguards. As shown in Table 20, forcing the agent to act rather than defer to the user causes the largest task-success degradation for Single-Turn and Static-Profile, indicating limited capacity to recover once an inappropriate autonomous decision enters the execution workflow. Compared with the strongest baselines, EASE improves task success by 11.7 percentage points in both domains and reduces consequential-action errors by at least 50.0%, while requiring fewer correction turns. Removing risk control raises the consequential action error rate by 3.3 and 5.0 percentage points in flight booking and meeting scheduling, respectively, demonstrating that preference adaptation alone cannot reliably prevent over-automation from propagating to consequential actions. Rather than relying solely on error-free autonomy, EASE preserves timely, low-cost intervention pathways for detecting and correcting misaligned decisions at the execution boundary.