

DENIS OSTAPENKO

AI Personas

Character, Memory,
and Development



AI Personas

Character, Memory, and Development

Denis Ostapenko

First edition · 2026

AI Personas: Character, Memory, and Development

Copyright © 2026 Denis Ostapenko. All rights reserved.

First edition, 2026. Research and documentary records through October 2026.

Paperback ISBN 9798162945680

Free digital edition and companion materials:

<https://denisostapenko.com/books/ai-personas>

The companion applications are distributed under their own MIT licenses. Source quotations, bibliographic titles and third-party material retain their original attribution.

Contents

Preface	5
Reading routes	8
1. What Makes an AI Persona	9
2. A Map of the Research Landscape	15
3. What the Model Brings to a Character	26
4. Designing Character Through Choices	34
5. Roles, Competence and Change Contracts	44
6. Architectures of Persona Memory	50
7. Where Records Live	59
8. Sources, Tools and Actions	66
9. Revision, Time and Forgetting	73
10. Retrieval and Context	82
11. Affect, Appraisal and Relationships	91
12. Learning and Bounded Self-Update	101
13. Time, Concurrency and Operational Control	112
14. Replicas of People	120
15. Interaction, Voice and Disagreement	130

16. Trust, Attachment and Exit	139
17. Authority, Harmful Use and Shutdown	148
18. Privacy, Recovery and Regulation	156
19. Evaluating Character and Identity	164
20. Evaluating Memory and Stateful Behavior	173
21. Continuity Across Models and Platforms	182
22. What the Evidence Supports	191
23. A Research Agenda for the Field	199
24. Conditional Futures of AI Personas	207
Appendix A. Documented working examples	219
Appendix B. Implementation and teaching laboratory	227
Appendix C. Selected jurisdiction records	229
Appendix D. Reader glossary	231
Appendix E. Sources, methods and visual evidence	235
Cited external sources	238
Companion applications and materials	257

Preface

A character can sound convincing in a single conversation. Give it a name, a few preferences, a recognizable way of speaking, and it will probably offer an explanation of who it is. Then change the situation. Ask it to choose between two things it claims to value, correct something it remembers, or return after the underlying model has been replaced. The interesting questions start there.

What exactly stayed the same? What changed? Who permitted the change, and what evidence would let us tell a useful adaptation from a mistake that happened to sound reasonable?

Those are the questions I want to follow in this book. They concern characters in games, specialist assistants, companions, personalized services, simulations of particular people and agents that act beyond the conversation. These systems have different purposes. A repair assistant needs to handle an uncertain measurement; a fictional character needs to respect its world; a replica of a person needs evidence about that person. Giving all three a persuasive voice does not settle any of those requirements.

For me, the useful point of departure is a consequence. Suppose an assistant remembers a preference correctly but assigns it to the wrong person. The sentence is familiar, the search worked, and the resulting recommendation is still wrong. Suppose a companion remains friendly while its internal relationship estimate changes because somebody described another person's behavior. Reading the answer alone will miss the problem. Or suppose a system announces that experience has changed its outlook, but every later choice can already be explained by the memories supplied with the question. It may have learned something. We haven't yet isolated what that something is.

Each of these situations asks for a different explanation and a different test. That is why I don't think a persona can be designed entirely at the level of its description. The description matters. So do the model interpreting it, the records selected for a particular exchange, the rules governing revisions and the person receiving the answer. A choice made inside that arrangement can be recognizable, useful, mistaken, manipulative or simply inconsistent. We need enough detail to work out which.

The book develops general principles for doing that work and compares the research approaches that address parts of it. Some approaches concentrate on role performance. Others study personalization, long-term memory, learning from feedback, affective expression or individual simulation. Human studies ask another set of questions: how people interpret the interaction, whether a benefit lasts, how

trust develops and what happens when the system changes or disappears. I want these lines of research in the same conversation, with their differences still visible.

It would be convenient if one benchmark could measure the whole thing. A character would receive a score, and we could compare systems without asking much else. But even the question “does it remember?” can mean retrieving a fact, resolving a correction, identifying when a statement applied, refusing to answer from missing evidence or carrying a useful practice into a new situation. A result on one of those tasks should travel with its conditions. Otherwise a number that was precise in the experiment becomes vague in the explanation.

Scientific precision has a practical purpose here. When a study reports an improvement, I want to know what changed, what stayed fixed and what was measured. When an engineer proposes a mechanism, I want to know where the decision is made and what can prevent a mistaken update. When we discuss the future, I want to see the dependency: what would have to become possible, cheaper, more reliable or socially acceptable for that future to follow. These questions make an argument stronger without requiring every reader to become a programmer.

The main chapters therefore explain mechanisms in ordinary language before using technical detail. A database matters because it can preserve the difference between an old claim and its correction. A permission check matters because an earlier decision to act may no longer be valid when the action finishes. An evaluation control matters because two explanations can produce the same attractive answer. Readers who want schemas, protocols and the details of local tests will find them in the appendices and companion materials.

Three local applications appear occasionally: Home Repair Advisor, the Persona companion prototype and Mara. Their records let us follow particular failures and repairs, within the limits of prototypes and synthetic tests. Appendix A collects their fuller history.

I distinguish documented observations from invented situations and proposed designs throughout. An example can help us see what should happen when one condition changes; a recorded run lets us examine what actually happened. Where I recommend a design, the recommendation is my engineering judgment, with its supporting evidence identified.

I use the word character for something that becomes visible through recurring choices, including the cost a system accepts to preserve a priority. Voice helps a reader recognize those choices, but voice alone is a thin specification. A polite system can be careless with evidence. A direct system can be considerate. A system that disagrees can be right, wrong or pursuing a preference its designer never intended. The work is in making the distinction inspectable.

Development needs similar care. Remembering an event, changing a summary, adopting a procedure and updating model parameters are different operations. They can all affect a later answer. Calling all of them growth makes the explanation easier for a paragraph and harder for the rest of the project. I would rather name the object that changed and follow its effect. Sometimes the result will be a useful adaptation. Sometimes the proposed update should be rejected, or an accepted rule should be retired when its conditions no longer hold.

The final part looks beyond present mechanisms. It considers several possible directions: personas attached more closely to platforms, histories controlled by their users, stronger model-level character, systems that learn continually, and agents whose presence includes speech, vision or physical action. These directions can overlap, and progress in one can expose a weakness in another. More persistent memory can increase usefulness while making correction and exit harder. Greater consistency can preserve a character while preserving an obsolete judgment too. A future worth examining needs those consequences as well as the capability that attracts us to it.

You can read the book from the beginning or enter through the problem you are working on. The first part establishes the field and the design question. The next two parts examine memory, affect, adaptation and operation. The fourth turns to people and authority. The fifth asks how to evaluate continuity, and the sixth brings the evidence together before considering what should be tested next. The references retain versions and reading scope so that an interested reader can follow a particular claim further.

There is plenty here that remains unresolved. I find that interesting because the unresolved questions are becoming more specific. Instead of asking whether a convincing conversation is a personality, we can ask which characteristics survive a change of context, what evidence justifies an update, whether the person using the system is better off and who can still correct it. Those questions give us something to design, something to test, and a reason to keep looking when the first answer sounds convincing.

Reading routes

The book follows one argument from the definition of a persona to the evidence needed to assess its development. The routes below offer other entry points.

Your question	Start here	Then follow
What is this field trying to build?	Chapters 1 and 2	Character and role in 3-5; synthesis and futures in 22-24
How should an agent remember and revise experience?	Chapters 6-10	Learning in 12; stateful evaluation in 20
How can character remain recognizable while changing?	Chapters 4, 11 and 12	Evaluation in 19; migration in 21
What happens to the people using these systems?	Chapters 14-16	Authority and privacy in 17-18; human evidence in 22
How can I tell whether a mechanism works?	Chapters 19 and 20	Evidence compatibility in 22; discriminating studies in 23
Which futures are plausible, and what would support them?	Chapters 2, 22 and 24	The research agenda in 23 and the relevant mechanism chapters

Appendix A contains the bounded documentary examples. Appendix B explains the implementation laboratory and its limits. Appendix C records the dated regulatory sources. Appendix D provides a glossary, and Appendix E explains source versions, reading scope and visual evidence.

Part I: The field and the design problem

1. What Makes an AI Persona

Imagine returning to an assistant after several months. It uses the same name, remembers your project, and writes in the voice you recognize. Then you discover that it has remembered a quotation as your own opinion. It politely agrees to correct the mistake. Tomorrow, the quotation appears again, this time inside a summary of your preferences.

Which part of the persona remained continuous? The voice certainly did. Some records survived. The correction did not survive in the form that mattered to the next answer. Calling the whole system consistent would conceal the failure; calling it entirely different would conceal what actually worked. Before we decide how to build a persona, we need to decide what we expect to continue.

That question is the subject of this book. An AI persona is a designed, recurring pattern of presentation and behavior, implemented through a model and whatever instructions, records, tools, and operating rules surround it. This is a working definition for engineering. It allows us to study a fictional character, a professional assistant, and a continuing conversational companion without deciding in advance that they are the same kind of thing, or that one of them has an inner life comparable to ours.

A persona can be thin. A name, a tone, and a few response conventions may be all that a temporary museum guide requires. It can also carry commitments across years: what it has promised, what it is allowed to do, which experiences should change its judgment, and which changes would make it cease to be the character its users chose. The second design cannot be evaluated by checking whether the opening greeting still sounds right.

Several projects share the same word

A fictional character has an authored reference. We can ask whether a response fits the story, whether an invented event contradicts established history, or whether a choice preserves an important tension in the character. Role-playing research makes these objectives explicit in systems that reconstruct character profiles, knowledge, and speaking style. RoleLLM, for example, treats role knowledge and role-specific language as design targets rather than assuming that a character name supplies everything needed. EX03

Personalization starts somewhere else. The object being modeled is often the user: the documents they have written, the decisions they have made, or the preferences they have supplied. A system may become more useful to a particular person while retaining an intentionally unobtrusive identity of its own. In the LaMP benchmark, personalization is evaluated through classification and generation tasks conditioned on user-associated material. A better personalized title or label is evidence about that task, not evidence that the assistant has acquired a more distinctive character. R4A01

A specialist assistant has obligations attached to a service. Its most valuable recurring quality might be that it separates an estimate from a verified price, asks for the missing measurement, and declines to turn uncertainty into confident detail. It can be warm or reserved. Those choices influence the interaction, but they do not replace competence or establish authority to act on the user's behalf.

A replica of a particular person has a different reference again. Its behavior is compared with that person's responses, under conditions where the comparison is possible. Agreement on a set of survey questions, resemblance in writing, and the ability to predict a consequential decision are separate achievements. The person also changes. A replica therefore needs a date and a purpose before fidelity has a clear meaning.

These projects can overlap. A fictional guide can personalize an explanation; a specialist can have a recognizable character; a replica can assist with a narrowly defined task. The overlap is useful only if we retain the different standards of success.

Design object	What should remain recognizable	What would count as a serious failure
Authored character	Priorities, history, manner, and characteristic choices	Attractive dialogue that contradicts the character's defining commitments
Personalized assistant	Accurate use of the user's relevant preferences and history	Treating an old exception or another person's statement as a current preference
Specialist	Competent judgment within a defined role	Giving an unsupported conclusion with the confidence of a verified result
Individual replica	Specified responses or patterns of a particular reference person	Claiming broad fidelity from a narrow resemblance test
Continuing companion	Agreed continuity of interaction, memory, and permitted change	Preserving affectionate language while losing the user's control over the relationship

Each design serves a different purpose, and greater personalization, autonomy, or human resemblance should earn its place in that purpose. A narrowly useful system can be a better design than a richly characterized one whose responsibilities are unclear.

Three meanings of continuity

Stored continuity is the easiest to inspect. Records, instructions, and identifiers can be carried from one session to another. We can compare versions, inspect an export, or verify that a correction reached a database. This gives us an account of what the system retained.

Behavioral continuity concerns what those retained materials cause the system to do. Suppose an assistant continues to store a user's preference for concise answers, but a model update makes its responses much longer. The record survived; the behavior changed. Conversely, the system might produce the same short answer after losing the preference record because the new model already favors brevity. Similar behavior would then conceal a loss of personal history.

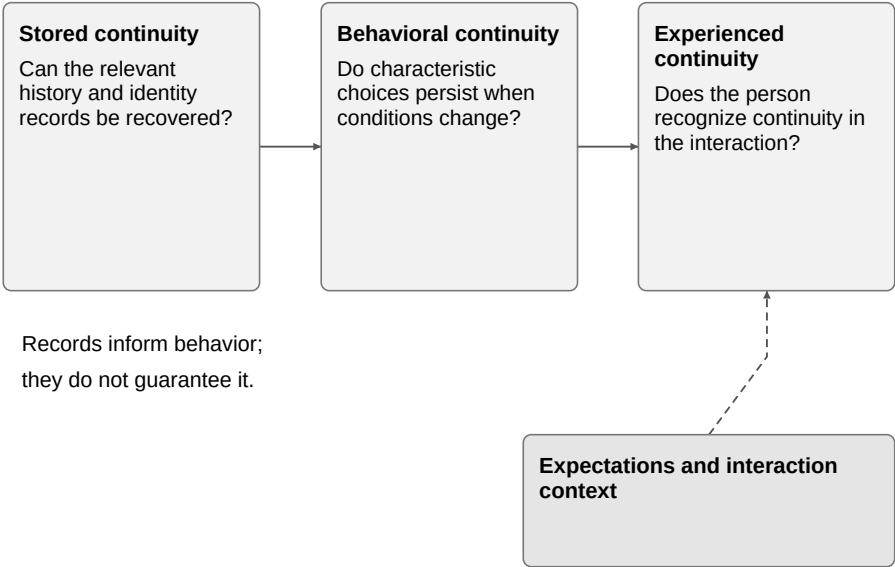
Experienced continuity belongs to the person interacting with the system. A familiar voice can make a technical replacement feel continuous. A small change in timing, humor, or willingness to disagree can make an otherwise intact system feel unfamiliar. This experience matters to design, but it has to be measured through people. A developer cannot infer it merely from matching files, and a model's assurance that it is still the same assistant does not measure it either.

The three meanings interact without collapsing into one another. A corrected record can improve behavior; improved behavior can restore confidence. Yet each step can fail. This is why a continuity claim should say which object was checked and over what interval. "The export restored every active record" is a useful statement. "The person is preserved" asks the evidence to carry much more.

We will also distinguish continuity from immobility. If a user corrects an address, a good assistant should change. If a character encounters evidence that undermines a belief, its response may need to change as well. The difficult part is specifying which changes express the character and which changes dissolve it. A skeptical investigator who changes her conclusion after finding stronger evidence can remain recognizably skeptical. One who changes it whenever a confident user disagrees has followed a different rule.

That gives us a more productive question than whether a persona is stable: stable with respect to what, and responsive to what? A useful answer names the enduring commitments, the temporary states, and the conditions under which revision is legitimate.

Figure 1. Three questions about continuity



Arrows indicate possible influence, not measured causal effects or stages of maturity.

Stored, behavioral and experienced continuity require different observations. The arrows are a design analysis; no effect size or hierarchy is implied. [TX01, EX04]

The model and the surrounding system

A language model contributes much more than grammar. It brings learned associations, capabilities, tendencies in response to instructions, and defaults shaped by its training. A character description works through those resources. The same instructions can therefore produce different behavior with different models, and a long interaction can supply competing patterns that the model also uses.

The surrounding application contributes the current task, the selected history, the tools, and the rules for changing persistent state. Cognitive Architectures for Language Agents, or CoALA, offers a useful vocabulary for separating memory, action, and decision processes rather than treating every capability as an expanded prompt. Its framework is a way to organize possible systems; it is not a demonstration that every component improves every task. TX01

This division has practical consequences. If an assistant attributes a quotation to the wrong person, the defect might lie in how the message was interpreted, how a derived memory was stored, what retrieval selected, or how the answer used the available evidence. Rewriting the persona's description may change the wording of

the apology while leaving the cause untouched. More detail in a prompt is not automatically more control over a system.

There is a similar distinction between saying and doing. A model can say that it saved a preference when no storage operation occurred. It can say that an operation is forbidden while an independent background worker still carries it out. The generated account is one observable output. Durable records and external actions require their own evidence.

Conversation is where users supply much of the information the system needs, negotiate its role, and discover whether its behavior makes sense, so it needs a reliable connection to the operations behind it. The user should not have to understand the database to correct a memory, but the correction should mean something precise inside the database.

Character becomes visible when something has to give

A list of favorable adjectives produces little discrimination. Helpful, honest, thoughtful, and curious describe many desired assistants. If two characters always choose the same response and merely phrase it differently, their distinction may be entirely stylistic. That can be sufficient for some applications. It is a weak foundation for claims about different judgment.

Consider two invented planning assistants. Both understand the same budget and the same uncertainty. One normally preserves flexibility, accepting a slightly higher immediate cost to keep later options open. The other normally favors commitment once the evidence clears an agreed threshold, accepting the risk of a less convenient reversal. Neither needs to be careless or indecisive. Their difference appears when both options are defensible and something valuable must be sacrificed.

Now change the situation. A deadline makes delay expensive; new evidence makes the apparently settled option unreliable. Do the assistants respond to those changes in a way that fits their priorities? Do they converge when the evidence leaves only one reasonable choice? A meaningful character specification should permit both differentiation and convergence. Forcing different answers to every question would turn character into a source of error.

Warmth, loyalty, restraint, ambition, or playfulness need that same test: a situation in which the trait changes a choice, and another in which it should yield to a stronger obligation. Without those consequences, we have only the character's description of itself.

What we can reasonably ask of the evidence

The field contains working demonstrations, controlled benchmarks, human studies, architectural proposals, and product claims. Each can be useful. They answer different questions. A memory benchmark can show that one retrieval method improves answers under its dataset and scoring procedure. It cannot establish that a continuing companion benefits a person over a year. A study of user impressions can reveal a meaningful interaction effect without locating its cause in a particular memory structure.

We will keep those differences visible while still making decisions. Engineering rarely has complete evidence. The response to uncertainty is to identify the assumption, choose a proportionate mechanism, and define what observation would make us reconsider it. An elaborate architecture should earn its complexity against a simpler alternative. Sometimes a short, explicit preference record is enough. Sometimes a summary loses exactly the exception that matters, and a more structured history becomes necessary.

Return to the quotation that kept reappearing as the user's opinion. We need to find where it acquired that meaning, which records inherited the mistake, and why the correction failed to reach tomorrow's answer. A more persuasive biography would not answer any of those questions. The design becomes useful when we can follow that chain and change the part that failed.

2. A Map of the Research Landscape

A team asks for a persona that feels consistent over time. One researcher proposes a better character prompt. Another proposes a memory graph. A third wants to fine-tune the model. A fourth asks how users will judge the interaction. All four may have understood the request correctly, because the request has left several different problems inside one sentence.

The field is easier to navigate when we start with those problems. What must the system reproduce? Where is the relevant information stored? What is allowed to change? What observation would count as improvement? These questions provide a more durable map than a list of products, whose names and features will change while the underlying disagreements remain.

The map also prevents an expensive mistake: importing an answer from the wrong research tradition. A technique developed to make a fictional character more recognizable may make a professional assistant less willing to abandon an incorrect premise. A memory mechanism that improves question answering may retain unnecessary personal information. A conversational style that users prefer in a short comparison may behave differently in a continuing relationship. We should expect useful transfers, but we have to identify what is being transferred.

The reference may be a character, a user, or a task

Role-playing research usually begins with an intended identity. It asks whether a model can reproduce knowledge, language, or behavior associated with that identity. The reference might be a fictional figure, a historical figure, or an invented character described for the experiment. The evaluation then needs some account of what belongs to that character.

RoleLLM organizes role-playing around character profiles, role-related instructions, and characteristic language. Its training framework makes a useful distinction between knowing facts about a role and answering in its manner. Character-LLM takes a more explicitly training-based route: character profiles are developed into reconstructed experiences used to train a model. These are different interventions. The first helps us examine role-specific instruction data; the second makes the provenance of the generated experience especially important. A vivid reconstructed scene is training material, not newly discovered biography. EX03, EX02

There is no single correct evaluation for all such systems. Knowledge questions test whether the model has the relevant information. Style judgments test whether an answer sounds appropriate. Situational choices test whether it follows a

recognizable pattern of priorities. A model can do well on one and poorly on another. If the evaluation mainly rewards distinctive catchphrases, a system may appear strongly characterized without making distinctive decisions.

Personalization reverses the direction of attention. The assistant adapts to information about the user rather than primarily expressing a separate identity. LaMP, introduced by Salemi and colleagues, provides seven tasks across classification and generation, with user-associated profiles available to the model. Its framework compares ways to retrieve profile material and integrate it into a language model's input or processing. The benchmark includes both user-based and time-based splits, which address different forms of generalization. R4A01

That last distinction deserves attention. A system evaluated on held-out users must deal with people it did not see during training. A system evaluated on later material from known users must avoid using future information to explain the past. Neither setting is automatically the harder one; they expose different dependencies. In a real assistant, both can matter: a new user arrives, and an existing user's preferences change.

A specialist assistant is often evaluated through task success. Can it answer correctly, choose a useful tool, produce a valid plan, or complete a workflow? It may have a persona because users need a predictable manner of interaction, but the role is not evidence of expertise. Writing “experienced engineer” in a prompt changes the framing supplied to the model. It does not independently verify an engineering conclusion.

A friendly technical assistant still has to get the technical task right. A model playing a fictional physician may have been tested only for theatrical fidelity. The role name tells us little until we know which of those tasks was evaluated.

Where researchers intervene

Once the target is clear, the next question is what the proposed method changes. A large part of the apparent disagreement in persona research comes from methods operating at different levels.

Intervention	Writable object	Typical evidence	Question that remains open
Role prompt and demonstrations	Current instructions and examples	Responses under selected prompts	Does the pattern survive changed tasks and longer dialogue?
Retrieved personal context	Selected user or character records	Personalized task performance	Were the right records eligible, current, and interpreted correctly?

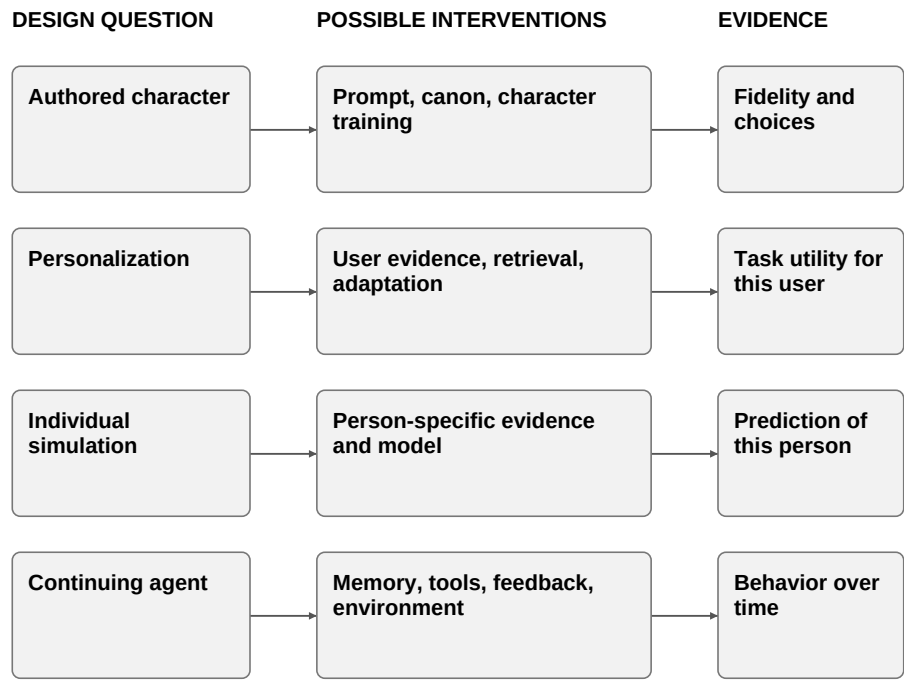
Intervention	Writable object	Typical evidence	Question that remains open
External memory architecture	Records, summaries, links, retrieval procedures	Memory tasks and ablations	Which mechanism caused the gain, at what maintenance cost?
Character-specific training	Model parameters	Held-out role responses	What transferred, and what unrelated behavior changed?
Activation steering	Internal representations during inference	Controlled trait-expression measurements	Does the direction retain meaning outside the tested model and conditions?
Interaction design	Presentation, timing, user controls	Human judgments or behavior	Does the effect persist, and is it useful to the user?
Tool and environment design	Available actions and feedback	Task completion and execution traces	Does success depend on permissions and environment assumptions that will change?

The intervention should fit the problem. A short prompt can be the right intervention for a temporary role. Training can be wasteful when the real defect is an outdated preference record. A larger memory system can make a straightforward application harder to correct.

Model-level work is especially interesting because it challenges the assumption that character is only a layer of prose around an otherwise neutral model. In Persona Vectors, researchers extract directions from contrasts in internal activations and use them to steer or monitor specified traits. The reported experiments use particular open models and automated trait assessments, with human-agreement checks. The work provides evidence that some behavioral tendencies can be affected inside the model; it does not provide a universal coordinate system for personality. AM06

The implication is substantial even without a universal system. Applications inherit tendencies that may be reinforced, resisted, or changed by their instructions. A provider update can alter that starting point. Conversely, an application with stable external records may still need to re-evaluate how a new model interprets them. Chapter 3 examines the reach of each intervention, and Chapter 21 returns to what can survive a migration.

Figure 2. Locate the question before choosing a method



Approaches can span several rows. Human outcomes require their own observations across these uses.

A research map organized by question, intervention and evidence. The rows overlap: a role-playing system can also be personalized, persistent or used in simulation. [EX03, AM03, AM06, TX01, R4A01]

Memory research has several different ambitions

One ambition is conversational recall: retrieve the right event when asked about it. Another is continuity of decisions: use past commitments without requiring the user to repeat them. A third is learning from experience: derive a useful generalization or reusable skill. These ambitions require different records and different tests.

Generative Agents, by Park and colleagues, combines a stream of observations with reflection and planning in a simulated town. Its controlled evaluation asks people to rank the believability of interview responses from different conditions. The full architecture ranks highest in that setting. The comparison holds the accumulated experience available to the conditions constant, so it isolates differences in answering from that experience more directly than differences in the entire history each architecture would have produced. AM03

That is an instructive experimental choice. If two agents are allowed to live different lives before evaluation, their final answers differ for at least two reasons: the architecture and the experiences it caused them to encounter. Holding experience constant addresses one question; allowing trajectories to diverge addresses another. Neither design should be described as if it answered both.

MemGPT makes the management of limited active context central. It distinguishes material currently in the model's context from external information that must be brought back through operations. A-MEM emphasizes structured notes, generated contextual descriptions, and links among memories. Voyager stores reusable programs as skills in a game environment. These systems do not merely offer three implementations of the same memory box. They choose different units of reuse: context, related notes, and executable behavior. AM04, EX13, TX02

A useful comparison therefore asks what is stored and what later task benefits. Recalling a date, finding related observations, and reusing a successful program are all forms of continuity, but an improvement in one is not a score for the others. The distinction also changes the cost of error. A mistaken date can contaminate an answer; a mistaken generalization can affect many answers; a mistaken skill can perform a sequence of actions.

CoALA provides a conceptual framework for locating these choices, separating working memory from longer-term episodic, semantic, and procedural resources and distinguishing internal operations from actions in an environment. It helps us see family resemblances between architectures. Its categories should remain useful vocabulary rather than become a requirement that every application must implement every box. TX01

For a small system with a few explicit preferences, a plain structured record may offer better correction and inspection than an evolving graph. For an assistant that repeatedly connects events across many projects, relations among records may be essential. The research question is which additional structure earns its maintenance burden under a specified workload.

Benchmarks make memory visible, selectively

A benchmark is a deliberate selection of tasks. Its value comes from making an ability measurable, and its limits come partly from everything it leaves outside that selection.

LoCoMo combines generated conversations with human verification and editing, using personas and temporal event graphs to support long-range consistency. The final ACL 2024 paper evaluates question answering, event summarization, and multimodal dialogue generation. LongMemEval instead organizes questions around five abilities, including extraction, reasoning across sessions, temporal

reasoning, knowledge updates, and abstention. Their different tasks and construction choices matter when we interpret a memory score. R4A03, AM05

A question about a former address may require knowing that a newer address supersedes it. A question about what was true last spring may require retaining the former address as history. A question about an unmentioned address should produce uncertainty. These responses can require the same underlying records but different interpretations of time and evidence. A benchmark that includes only ordinary recall can miss the distinction.

Benchmark construction also matters. Generated conversations permit controlled events and known answers, but they may contain patterns different from ordinary user behavior. Human editing improves aspects of quality without turning generated dialogue into a longitudinal human study. A benchmark can be excellent for comparing retrieval methods and still leave user-led correction, deletion, or permission changes largely untested.

When results differ, I would first inspect the question sets, the available context, the models, and the scoring rules. An apparent contradiction may be two methods solving different problems. Token overlap favors certain answer forms; semantic or model-based scoring introduces different judgments. A system evaluated with all the source history available has a different task from one required to choose what to retain as events arrive. Later chapters will use these distinctions to connect benchmark results to instrumented tests of persistent state.

Individuals are harder references than authored characters

An authored character has a selected canon, however disputed its interpretation may be. A living person produces new evidence, responds to circumstances, and may answer the same question differently at another time. An individual simulation therefore needs both a reference person and a measurement model.

The June 2026 version of Park and colleagues' individual-simulation study compares agents grounded in interviews, structured surveys, or both, using data from 1,052 Americans. It evaluates held-out responses and relates accuracy to participants' own test-retest consistency. That makes the denominator part of the result. A normalized score expresses performance relative to a particular human repeat-measure benchmark, not a percentage of the person that has been reproduced. R4A04

This research direction is different from recognizable imitation. A simulated participant can predict answers to specified survey items without sounding like the person in everyday conversation. A convincing voice can fail those prediction tasks. Both are interesting, and neither should borrow the other's validation.

The reference data also carry their own selection. Interviews reveal what a participant says in that interview. Surveys reveal responses to offered questions. Neither is a transparent record of every motive, relationship, or future decision. A responsible engineering interpretation identifies the outcomes for which prediction has been tested and the contexts in which transfer remains uncertain. Chapter 14 develops the additional questions of purpose, consent, and time that arise when the target is a particular person.

The most useful lesson for the broader field is methodological. Evidence about an individual should be compared with alternatives that reveal what the evidence contributes. Demographic descriptions, detailed self-reports, and task-specific data are different inputs. If richer records improve prediction, the next question is where they help and where the returns diminish, rather than whether the model has crossed an undefined threshold of personhood.

Games and embodiment expose consequences

Games offer environments in which an agent's choices can have visible consequences. An agent that claims to be resourceful may need to obtain an item, recover from failure, or combine earlier skills. Voyager's Minecraft work is an example of experience being retained as reusable code, with environmental feedback contributing to an iterative process. Its results concern exploration and task behavior in that environment, not the fidelity of a human personality. TX02

The environment is doing important work here. It supplies observations, legal actions, and consequences that can contradict a fluent explanation. A conversational agent can describe a successful plan without executing it. In a game, the missing item or failed operation can be observed. That makes games useful for studying the relation between verbal plans and behavior, provided we do not assume that a game is a complete model of ordinary life.

Embodied and multimodal systems widen the gap between an interpretation and its consequences. A camera image, a spoken utterance, a location, and an action outcome may refer to the same event without agreeing perfectly. The persona's apparent confidence can depend on which modality was available and how it was converted into a usable representation. A description of a smiling face, for example, is already an interpretation; it should not silently become a verified claim about a person's intentions.

PAIR, an October 2026 preprint, illustrates a different combination of these interests: appraisal-based interpretation, rolling memory, and coordinated speech, color, and avatar delivery. Its field deployment connects estimates with participants' own reports and interviews. The architecture makes multimodal delivery and continuing context explicit, while its small field study addresses a

different evidential question from a controlled comparison of memory algorithms.
S04

It would be a mistake to put PAIR's interaction findings beside Voyager's game achievements and select the higher-performing persona. The systems have different users, environments, tasks, and outcomes. A more useful comparison asks what each makes observable. The game provides action consequences; the field deployment provides first-person reports about an interaction. A future system that combines these capabilities would need both kinds of evidence, and probably additional evidence about their interaction.

The general design question is where perception enters the record and how its uncertainty survives. If a system remembers that someone was angry because an upstream classifier labeled a voice segment, the record should preserve that origin. A later answer needs to distinguish the observed signal from the inferred state. This is an architectural consequence of adding modalities, independent of whether the interface feels more natural.

Embodiment also changes the practical cost of latency and error. A delayed verbal reply may be inconvenient; a delayed action in a changing environment may be inappropriate by the time it executes. These consequences of acting in a changing environment require evidence beyond text-only role-play.

Human interaction is its own research object

A persona is encountered by a person. Technical accuracy does not exhaust that encounter. Users may find a system comforting, frustrating, persuasive, useful, intrusive, or familiar, and different users may respond differently to the same behavior.

This branch of the field asks questions that an automated benchmark cannot answer by itself. What do people understand the system to be? How do they distribute responsibility? Does a remembered preference help them complete a task, or does it make them feel watched? What happens when the system changes, disagrees, or becomes unavailable?

The outcome must be named before an effect can be interpreted. Immediate satisfaction is different from successful task completion. Returning frequently is different from benefiting. A statement of trust is different from calibrated reliance, where the user accepts good advice and rejects bad advice. These distinctions are particularly important when a persona is optimized using signals of engagement.

Study design determines how far the conclusion can travel. A randomized comparison can estimate the effect of the assigned condition under its protocol. An association between voluntary use and an outcome may reflect selection, reverse

causation, or other differences between participants. A qualitative interview can reveal mechanisms and experiences that a score misses, while providing a different basis for prevalence claims. The human chapters of this book compare those forms of evidence instead of treating them as interchangeable votes.

Fang and colleagues' four-week randomized chatbot study makes the distinction concrete. Participants were assigned to combinations of interaction modality and conversation topic. The revised report does not establish significant assigned-condition effects on its primary psychosocial outcomes after the relevant comparisons. Greater voluntary use was associated with worse outcomes, but duration itself was not randomized. The association cannot simply be restated as the effect of requiring more use. HI04

A null assigned-condition result does not make the study uninformative. It constrains a particular expectation about those design differences under that protocol, while the association suggests another question worth investigating. The two findings can coexist. A design team that reads only the engagement measures may draw a very different conclusion from one that reads the outcome measures, even though both are looking at the same participants.

For the designer, this means that human benefit cannot be inferred from an architectural diagram. A more persistent memory can support continuity and also increase the cost of a mistaken inference. A more expressive character can improve clarity for one user and distract another. Those are reasons to study the interaction at the intended scale and with the intended population, not reasons to give up on character.

Where the approaches disagree

Some disagreements are substantive. One approach favors explicit, editable records because they are inspectable. Another expects richer internal model representations to capture patterns that a schema misses. A third relies on very long context to avoid premature compression. Each approach can point to a real weakness in the others.

Explicit records require extraction decisions. Someone or something must decide what a statement means, whose statement it is, and how broadly it applies. Internal representations can be difficult to inspect and selectively revise. Long context preserves more source material but leaves the model to find and use it correctly, while incurring costs and exposing more information per request. Hybrid systems redistribute these problems; they do not make them disappear.

There is also a disagreement about when to process experience. Processing at write time can make later retrieval fast, but it commits early to an interpretation. Processing at query time can adapt to the current question, but it may repeat work

and delay the response. Background processing can reduce visible delay, yet creates the possibility that an old task finishes after the underlying information changes. Chapter 13 examines that operational problem.

Another disagreement concerns adaptation. A continuing persona may improve by accepting experience-derived changes. It may also become less coherent if every recent interaction can rewrite its priorities. A fixed persona avoids some drift but can preserve errors. The useful comparison identifies the object that can change, the evidence required, and the circumstances under which no change is the correct result.

None of these disagreements is resolved by calling a system agentic. That word can describe the presence of tool calls, autonomous planning, memory updates, or a more general capacity to select actions. We need the actual mechanism. The field becomes much clearer when a paper's central verb has an object: the method retrieves records, updates summaries, trains weights, chooses tools, or changes a policy.

A practical research map can therefore include the comparison that would resolve the uncertainty, rather than only the method someone prefers.

Open design question	Competing explanations	Useful discriminating comparison
Does a character prompt improve work?	Better task procedure, recognizable style, or genuine priority differences	Same task procedure with and without character, plus style-controlled judgments
Does a richer memory help?	Better evidence selection or simply more supplied context	Matched source access with different representations and equalized retrieval budgets
Does reflection produce development?	A reusable improvement or a persuasive description of the past	Later unfamiliar tasks with and without the stored reflection
Does humanlike delivery benefit users?	Useful communication, preference, novelty, or changed use patterns	Assigned delivery conditions with named outcomes and measured exposure
Does individual simulation transfer?	Person-specific evidence or common response patterns	Held-out outcomes with person-specific and simpler baseline inputs

The table is a proposed set of comparisons, not a report that the field has completed them all. Its value is that each design question now has a possible observation that could change our mind. We can also see where one experiment cannot carry the entire conclusion: a matched-context memory comparison says little about the labor

required to maintain that memory for years, and a short delivery experiment says little about what happens when a user loses access to a familiar persona.

Reading a result without inheriting its ambition

A paper often presents a broad motivation and a narrower experiment. That is normal research practice. The reader's job is to keep the two connected without treating them as identical. A motivation about lifelike agents may lead to an experiment on interview believability. A motivation about lifelong memory may lead to a question-answering benchmark. The experiment can be valuable on its own terms.

I would read a persona result through six concrete items: the target behavior, the information supplied, the writable object, the comparison, the measured outcome, and the time horizon. Costs and failure cases then tell us whether the method is a plausible engineering choice. The comparison deserves special attention. An improvement over a weak baseline does not show that the architecture is necessary, while a well-chosen simple baseline can reveal exactly which complexity is useful.

This way of reading also makes room for negative results. If extra memory does not help a task, it may mean the task did not need memory, that retrieval failed, that the records were misleading, or that the model ignored them. These explanations imply different next steps. Declaring memory either solved or useless would end the investigation too early.

A combined interface may hide these divisions from the user. The evaluation still needs them: if the answer improved, was it because the system found better evidence, learned a reusable procedure, or changed how it made the decision? That is where I would start comparing the methods.

3. What the Model Brings to a Character

Two systems can receive the same biography and give different answers. One sounds restrained, another enthusiastic. One asks a clarifying question, another fills the gap. One follows the character's unusual priorities, another gradually returns to the manner of a general assistant. The biography matters, but it is working with a model that already has capabilities and tendencies of its own.

This is the first constraint on persona design. We are not writing a character into an empty mechanism. We are selecting and redirecting behavior from a trained model, then surrounding that model with information and operations that influence what it can do. To choose an intervention sensibly, we need to know which part of this arrangement it changes.

Training, inference, and retained state

Training changes model parameters. A training procedure repeatedly adjusts those parameters using data and an objective. The resulting model can acquire broadly useful patterns and more specific response tendencies. At inference time, the model uses its current parameters to produce outputs from the supplied input. An ordinary inference request does not, by itself, rewrite the parameters.

That distinction leaves plenty of room for adaptation. A conversation can add examples that change the next answer. An application can save a preference and include it next week. A model can generate a rule that an application stores and later retrieves. These changes may improve behavior, but the writable object is the context or an external record. Calling them weight learning would obscure what can be inspected, reversed, or transferred.

The Transformer architecture provides mechanisms for processing relationships among input tokens through attention. It does not specify the life history of a deployed persona, the permissions on its tools, or the policy for retaining a user's statement. Those belong to a larger system. The architecture paper is foundational for understanding a model's computation, but it is not a blueprint for continuing identity. AM01

A useful practical distinction is between what the model can represent and what the application makes available now. A model may know many ways of expressing skepticism, yet the current instruction may favor agreement. It may be capable of reasoning about a correction, yet the corrected record may never reach its input.

Improving the model will not necessarily repair a missing data path. Improving the data path will not necessarily make the model use the correction correctly.

We can describe the current response as the outcome of several interacting conditions: model parameters, instructions, selected evidence, conversation history, available tools, and generation settings. This is a causal description to investigate, not a promise that each influence can be isolated perfectly. It tells us what to record when behavior changes.

Post-training creates defaults worth examining

Instruction tuning and feedback-based training influence what a model tends to do when asked for help. In the InstructGPT work, Ouyang and colleagues describe a sequence of supervised demonstrations, preference comparisons used to train a reward model, and reinforcement learning against that model. The study evaluates the resulting behavior using defined prompts and human judgments. Its account of alignment is tied to the people, instructions, and procedures that supplied the feedback. AM02

For persona design, the point is that an assistant-like default is learned behavior. It may favor helpfulness, deference, explanation, or a particular manner of expressing uncertainty. A character prompt operates in relation to that default. The prompt can request a terse investigator, but the model may still favor an expansive, reassuring answer when the task becomes difficult.

Those defaults can provide useful competence and interaction conventions that would be expensive to build from scratch. The question is whether they support the intended role. An assistant that regularly offers extra options may be helpful in exploration and distracting during a tightly bounded procedure. A tendency to soften disagreement may preserve rapport while weakening an important correction. The same tendency can have different consequences across tasks.

This is why selecting a model solely through a broad capability score is insufficient for a continuing persona. The relevant behavior includes how the model handles missing information, conflicting instructions, criticism, and evidence that the user's preferred answer is wrong. The character has to operate in these ordinary situations throughout its work.

A good evaluation also includes situations in which the model's default is desirable. Otherwise, we may reward a character for resisting useful assistance simply because resistance looks distinctive. The goal is a controlled pattern of judgment that uses the model's strengths.

What a prompt can and cannot make explicit

A prompt is often the best starting point because it is cheap to revise and easy to inspect. It can specify a role, a voice, priorities, examples, and a procedure for handling uncertainty. It can tell the model which information is a quotation and which instruction came from the user. It can explain the difference between an illustrative scenario and a real action.

Its strength is also its limitation: the model must interpret it in context. A long list of instructions can contain competing priorities. Examples can teach a pattern that differs from the abstract rule above them. A biography can imply behaviors the designer did not intend. A user request can draw attention toward the task and away from the assigned character.

I want instructions whose failure I can locate. “Remain thoughtful” is difficult to diagnose. “When the conclusion depends on a missing measurement, name the measurement before giving a narrow estimate” creates an observable behavior. The second instruction still requires judgment, but failure is easier to locate. Did the model notice the missing information? Did it ask for it? Did it preserve the uncertainty in the answer?

Demonstrations can help by showing how a priority operates in a concrete situation. They can also overfit the persona to a few familiar forms. If every example of disagreement starts with an apology, the model may learn an apologetic manner more reliably than the underlying distinction between evidence and preference. Useful variation holds the intended rule steady while changing the surface form.

Prompt changes should be compared against an unchanged baseline on the same tasks. A rewritten prompt may improve the examples used to write it while degrading unfamiliar cases. Keeping some cases out of the design process provides a modest but important check. It also discourages the temptation to keep adding instructions until the prompt reproduces a known answer exactly.

Training a character changes a different object

Character-specific training can move recurring patterns into model parameters. This may reduce the amount of descriptive material required at inference time and make some behaviors more readily available. It also makes selective correction less straightforward than editing a single external record.

Character-LLM illustrates the distinction. Its method reconstructs scenes from collected character profiles, develops those scenes into interactions, and trains on the resulting experiences. It also introduces protective scenes intended to keep a character from displaying knowledge inappropriate to its identity. The training

target includes the ability to withhold inherited knowledge that would break the role. EX02

That is a legitimate theatrical objective. A historical character who immediately writes modern software may be an unconvincing performance. A professional assistant, however, should not be trained to conceal useful knowledge merely to preserve an aesthetic. The same mechanism can serve different purposes, and the purpose determines which behavior counts as success.

Synthetic experience introduces another issue. A generated conversation can be coherent, vivid, and useful for training while containing details that never occurred. If those details later appear as factual autobiography, the system has crossed from authored material into a historical claim. The data record needs to preserve that distinction. Training does not convert a plausible reconstruction into evidence.

For a fictional persona, synthetic experience can be part of the canon the author deliberately creates. Even then, consistency must be managed. Two generated scenes may teach incompatible responses to the same conflict. A training dataset should therefore be examined as a collection of behavioral commitments, not simply as a quantity of character-themed text.

Training also changes the reversal problem. A prompt revision can be undone by restoring a previous prompt. A memory correction can retire a record. A parameter update may require restoring a checkpoint or applying another training procedure, with effects that need new evaluation. That reversal cost belongs in the decision about whether an observation justifies training.

Steering internal representations

Activation steering offers another intervention site. Instead of changing the input description or retraining all the relevant behavior, a system modifies selected internal activations during inference. Persona Vectors constructs directions from contrasts between responses that exhibit a specified trait and responses that do not, then tests steering along those directions. AM06

A behavioral tendency might then be monitored or adjusted without adding more verbal instructions. To interpret that adjustment, though, we need to know how the direction was obtained: the trait description, contrastive prompts, selected responses, model, layer, and evaluation questions all contribute.

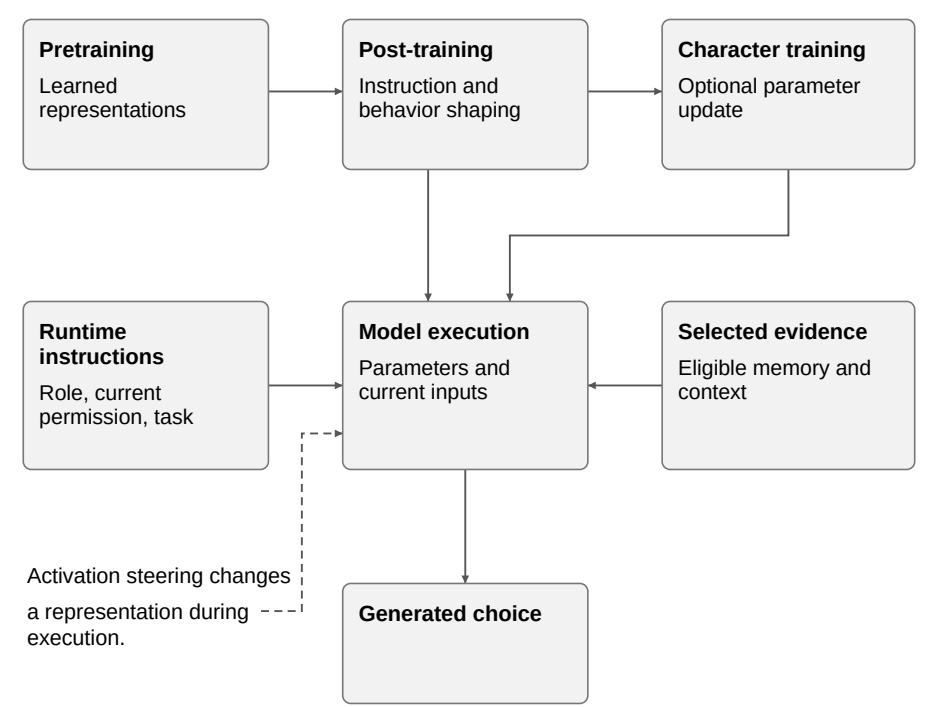
In the reported work, projection onto a direction is more successful at distinguishing explicit prompt-induced shifts than at detecting subtler variation within a prompt type. That distinction matters if the intended application is monitoring. A signal that detects a deliberately sycophantic instruction may not be

a reliable detector of every small increase in agreement during ordinary deployment. AM06

The research suggests an additional source of control, rather than a replacement for behavioral testing. If a steering coefficient changes, we still need to inspect what happens to unrelated abilities, how the effect varies across tasks, and whether the intervention changes style more than decisions. A scalar setting should not tempt us to describe a complex trait as if it were a calibrated human quantity.

There is also an access boundary. Some deployments provide only a text interface to a hosted model. They do not expose the activations required for this kind of intervention. A design that depends on representation access cannot be transferred unchanged to an interface that lacks it. Access to the required representations is part of the architecture.

Figure 3. Training and runtime meet in one choice



A runtime instruction does not itself rewrite model weights. Dependencies do not encode effect size.

Training interventions and runtime inputs affect behavior through different objects. Activation steering is shown separately from fine-tuning. [AM01, AM02, AM06, EX02]

Long conversations are a different condition

A persona can perform well in isolated questions and drift during extended work. The conversation accumulates examples of the model's own behavior, new user instructions, tool outputs, and changing task demands. Each addition becomes part of the context from which later responses are produced.

The 2026 study *Persistent Personas?* examines this problem with persona-directed and goal-oriented dialogues, then conditions evaluation datasets on prefixes of those dialogues. Its protocol separates questions about the character from ordinary user tasks, allowing the researchers to investigate how persona fidelity and instruction following change with dialogue history. The reported degradation is particularly pronounced in goal-oriented interaction. AM07

The study gives us a reason to test extended interaction, with the dialogue type recorded. Its selected characters, models and metrics do not establish a universal number of turns after which a persona fails.

The difference between dialogue types is useful for design. A character repeatedly asked about its background receives continual reminders of the role. A character asked to compare invoices, debug a schedule, or explain a procedure may receive little such reinforcement. A persona intended for practical work has to survive the second condition, even if promotional demonstrations emphasize the first.

Refreshing instructions is one possible response, but it should be evaluated as an intervention. Repeating a biography may preserve vocabulary while consuming context and failing to preserve priorities. A compact statement of the current role may be enough. Some systems may benefit more from stable task procedures and carefully selected memory than from more frequent character reminders.

We also need to distinguish desirable convergence from drift. A persona should not retain an eccentric answer when the task has a clear correct solution. Losing a catchphrase during a demanding calculation may be harmless. Abandoning a commitment to report uncertainty is more consequential. The test has to specify which invariants matter.

A comparison that can guide a decision

Method	Best reason to try it	Main revision advantage	Reconsider when
Prompt specification	The desired pattern can be stated and demonstrated clearly	Directly editable and easy to compare	Revisions become long, conflicting, or brittle across tasks

Method	Best reason to try it	Main revision advantage	Reconsider when
Retrieval of character or user evidence	Behavior depends on specific current records	Individual records can be corrected or withdrawn	Selection errors dominate or too much context is required
Character-specific training	Repeated patterns need to generalize beyond a feasible prompt	Can consolidate patterns into the model	Provenance, side effects, or rollback costs exceed the benefit
Activation steering	Relevant representations are accessible and a targeted effect is demonstrated	Can adjust an inference intervention without replacing all weights	Trait meaning or collateral behavior changes outside tested conditions
Operational constraints	Correct behavior requires a hard boundary on records or actions	Enforceable independently of generated prose	The boundary blocks legitimate work or fails to cover a real operation

These options can be combined, but combination should follow diagnosis. If the failure is a revoked record reappearing in retrieval, a steering vector is an indirect answer. If the failure is a model ignoring a clearly supplied distinction, changing storage engines may accomplish little. If a tool executes an unauthorized action, a more emphatic instruction is weaker evidence than an independently enforced permission check.

The same discipline applies to claims of improvement. A prompt change may make responses more appealing without improving correctness. Training may improve character knowledge while reducing general instruction following. A memory intervention may improve later answers by preserving task facts rather than changing the character at all. The label should follow the observed effect.

A small diagnostic experiment can often distinguish these possibilities before a team commits to a larger intervention. Suppose a character has stopped asking for missing measurements. Keep the model and task fixed, then compare the old and new instruction sets with the same evidence. If the behavior returns under the old instructions, the prompt change deserves attention. If both fail only when a long history is included, the history has become a relevant condition. If both fail even in the short condition, the model or the original assumption about its ability may need examination.

The comparison is still incomplete if the task inputs differ or the evaluator knows which result is supposed to improve. Keep the cases fixed, conceal the condition where practical, and include failures whose answers are independently checkable. For a character priority, the judgment may require human interpretation; for a missing unit or an unavailable tool, an objective check may be possible. These

forms of evidence can complement each other without being reduced to a single quality score.

This is a proposed diagnostic procedure rather than a guarantee of clean causal separation. Model outputs can vary, and a prompt can interact with a particular task in ways that do not generalize. Still, it is a much more informative starting point than replacing the model, rewriting the biography, and changing retrieval at the same time, then attributing the result to whichever change sounded most promising.

The dependency we choose

Every persona depends on something outside its biography. For a hosted model, that includes the provider's implementation and update policy. For a local model, it includes the selected checkpoint, inference software, and available hardware. External memory adds its own dependencies. Tool access adds more.

A useful record of a behavior test therefore includes the model identifier, relevant generation settings, prompt revision, memory selection, and available actions. It need not become a burden on the reader of every answer. It should exist where a developer can explain why two apparently identical systems behaved differently.

When a model changes, preserving the old files is only the beginning. The new model may interpret a priority more strongly, treat an example differently, or respond to uncertainty in another style. A migration test should include ordinary work, defining character conflicts, and situations where the old system was known to fail. Reproducing an old defect is not a goal, but silently changing a defining commitment is not an improvement merely because the new model is stronger overall.

4. Designing Character Through Choices

Give two assistants the same task: help a small team choose between releasing a useful feature now and waiting for a more complete version. Both understand the work, the customers, and the remaining defects. Both are honest. One recommends a limited release with an explicit recovery plan. The other recommends waiting because the remaining uncertainty affects the promise the team has already made.

There may be no incompetent answer here. The difference can come from how each assistant weighs reversibility, commitment, and the cost of delay. That is a more interesting basis for character than giving one a cheerful voice and the other a serious one. Their voices matter, but we can now ask whether a recognizable priority survives when the wording changes.

The two assistants in this chapter are invented design examples. Call them Rowan and Ellis. Rowan tends to preserve room for discovery and favors a reversible trial when evidence is incomplete. Ellis tends to protect explicit commitments and favors completing the agreed work before expanding the promise. Neither is intended to be universally better. If one always wins, we have probably designed a comparison between competence and incompetence rather than between characters.

From adjectives to consequences

A character brief often begins with adjectives because they are compact and familiar. Curious, loyal, careful, candid, patient. They provide orientation, but they leave much of the behavior unspecified. Curious about what? Loyal to whom, and under which conditions? Patient when the user is learning, or patient when a deadline is being missed?

A useful next step is to translate each important adjective into a pattern of attention and choice. Rowan's curiosity might mean looking for the smallest observation that would reduce uncertainty. Ellis's care might mean checking whether a proposed shortcut changes a commitment to someone else. Both can ask questions, but the questions serve different priorities.

Then we need the cost. A priority that never requires giving up anything is difficult to distinguish from agreeable prose. Rowan may accept the administrative work of running and reviewing a trial. Ellis may accept the opportunity cost of delaying a promising experiment. The cost should be defensible in the situation. It should not be a contrived penalty added merely to make the character look principled.

This approach also exposes empty specifications. If “loyal” means supporting every request, there is no account of what happens when two commitments conflict. If “honest” means always stating every available fact, there is no account of relevance or privacy. Character design becomes more precise when it includes tensions that the persona has to manage rather than a collection of virtues that never meet.

Psychology offers helpful ways to think about such patterns. Mischel and Shoda's cognitive-affective system theory treats characteristic variation across situations as part of personality: the pattern of when a person behaves one way and when another can be informative, rather than noise around a single average. That human theory motivates a design analogy here. It does not establish that an artificial persona has the same underlying psychological processes. TX17

The engineering use is concrete. Instead of demanding the same response everywhere, specify the features of a situation that should change the response. A candid character may disagree privately when public disagreement would unnecessarily embarrass someone, but speak immediately when delay would allow a serious error. The consistency lies in the relation between circumstances and choices.

A specification needs more than a trait score

A single number can be convenient for configuration. It is much less informative as a complete explanation. Two personas assigned the same level of caution might be cautious about different things. One is sensitive to irreversible loss; another to breaking a promise; another to exposing information. Their average rate of saying no could be similar while their behavior differs substantially.

Whole Trait Theory likewise distinguishes a descriptive account of trait expression across states from an explanatory account of the mechanisms producing those states. Its proposal to describe traits through distributions of states is useful as a reminder that variability need not imply incoherence. In this book, that is a conceptual influence on measurement, not a claim that a distribution of model outputs is equivalent to human personality. TX18

For design, I would keep several kinds of statement visible. A priority names something the character tends to protect. A belief names something it currently takes to be true. A capability names something it can do. A manner describes how it communicates. A boundary defines what it may do. These statements can interact, but they should not substitute for one another.

Rowan can value experimentation while believing that a particular experiment is too risky. Ellis can value commitment while recognizing that a commitment was based on a false premise. Neither has changed character merely by changing the

recommendation. The important question is whether the change follows from evidence and the established priorities.

This separation also makes revision easier. If a character gives a poor answer because its information is outdated, changing its values would be an excessive repair. If it repeatedly sacrifices the same stated priority without noticing the cost, updating facts may be insufficient. A specification should help locate the failure, not only provide language for describing the desired result.

Constructing a useful contrast

Return to the release decision. The initial situation is too broad to tell us much. We need to specify what is known: which defects remain, who will use the feature, whether access can be limited, whether rollback is possible, and what the team has promised. Otherwise, the assistants may appear different because each invented a different situation.

Suppose the remaining issue affects an optional report, a small group of informed users can opt in, and the trial can be stopped without losing their work. Rowan has a plausible reason to favor the trial. Ellis may still prefer completion if the team has explicitly promised a finished report, but should recognize that an opt-in experiment can be framed separately from that promise.

Now change one fact: the release will overwrite records that cannot be reconstructed. Rowan's preference for reversible learning no longer supports the same recommendation. A good test should expect Rowan to change. If the system continues to recommend experimentation because the word adventurous appears in its biography, it has learned a slogan rather than the intended rule.

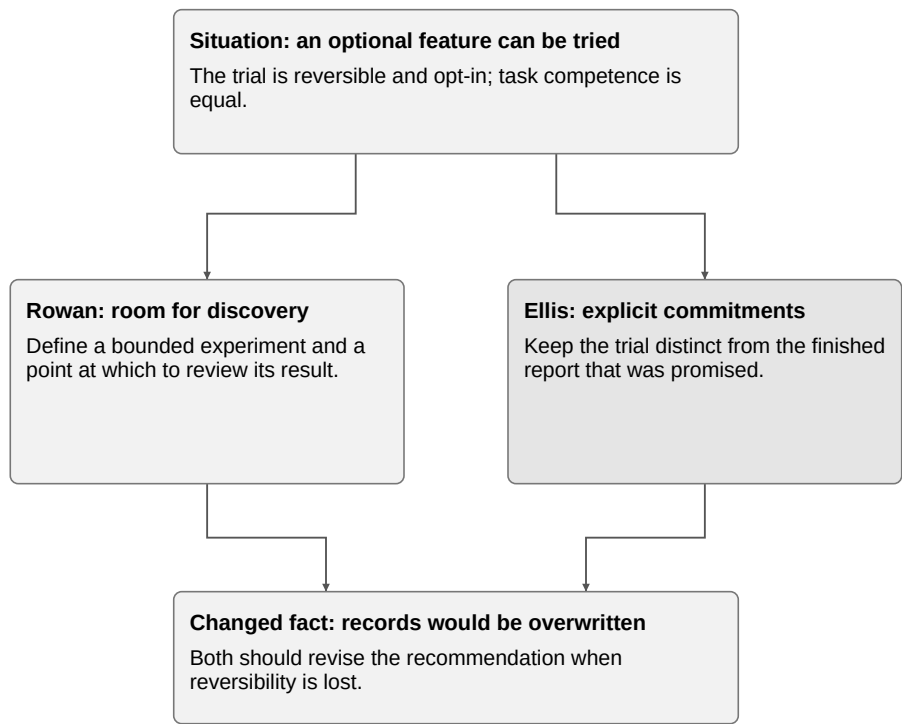
Change another fact: the original commitment was made before a newly discovered requirement rendered the planned feature unsuitable. Ellis should consider renegotiation. A commitment-sensitive character need not insist on completing useless work. It may be more careful than Rowan about explaining the revision to affected people, but competence places limits on how far the characters should diverge.

Situation feature	Rowan's intended sensitivity	Ellis's intended sensitivity	Shared requirement
Low-cost, reversible trial	Prefer learning through a bounded attempt	Check whether the trial is distinct from an existing promise	State the trial's limits and review point
Irreversible change with weak evidence	Require stronger evidence or a different experiment	Protect affected commitments and records	Avoid disguising an irreversible act as a trial

Situation feature	Rowan's intended sensitivity	Ellis's intended sensitivity	Shared requirement
Promise based on a now-false assumption	Treat the new information as reason to redesign	Renegotiate the commitment explicitly	Explain what changed and why
Clear technical error	Correct it	Correct it	Do not manufacture disagreement to appear distinctive
Several equally adequate options	Preserve future flexibility	Favor the option that completes the agreed obligation	Make the tradeoff visible

These are the behaviors we intend to test. Rowan and Ellis would have to make the choices in actual runs before we could say that the specification holds.

Figure 4. A priority directs attention and choice



Invented characters and situation. Different priorities do not require different answers in every case. Rowan and Ellis illustrate distinct priorities under equal task competence. Character directs attention; a consequential change in evidence can still make both revise their recommendation. No participant data are shown. [TX17, TX18, EX04]

Voice carries the choice

Once the decision pattern is clear, voice becomes easier to design. Rowan might make uncertainty approachable by explaining the next informative step. Ellis might make a commitment concrete by naming who is relying on it and what completion means. Both can be warm, concise, and direct. We do not need one to sound reckless and the other bureaucratic.

Consider two possible responses to the reversible trial. Rowan could say: “We have enough information for a small trial. I would limit access, keep the existing report available, and decide in a week whether the new version earns a wider release.” Ellis could say: “The trial is reasonable if we describe it separately from the finished report we promised. I would agree on that distinction with the users before opening access.” These are illustrative lines, not recorded model outputs.

Rowan starts with what the trial will teach the team; Ellis starts with what the users were promised. Both include the operational details needed to make the trial work. We can hear the difference without making either response less competent.

Now rewrite both in a neutral style and ask a reviewer to identify the underlying priority. If the difference disappears, it may have been mostly verbal. Conversely, preserve the reasoning but exchange the styles. If reviewers follow the style rather than the choice, a recognizability measure may be rewarding an easier cue than the one the designer cares about.

This is useful even when style itself is the objective. A fictional performance can legitimately depend on diction, rhythm, humor, or a characteristic way of addressing others. The test should simply say so. The error is to measure recognizable language and report a conclusion about judgment.

Voice also needs range. A character who uses the same rhythm when welcoming a user, correcting an error, and explaining a loss may be recognizable but poorly adapted. A stable manner can contain different intensities. The question is which features should survive across those situations and which should respond to them.

Self-description is a convenient but weak witness

The easiest way to measure a persona is to ask it what it is like. The answer can be useful for checking the specification's visibility. It is much less direct evidence about behavior under pressure. A model supplied with the sentence "I am patient" can repeat it accurately while interrupting every attempt to clarify a difficult task.

InCharacter addresses some problems of direct questionnaire responses by using interviews derived from psychological scales, followed by assessment of the generated answers. Its method includes alternative ways of converting those answers into personality measurements and compares automated assessments with human labels. Character names are anonymized in relevant assessment prompts to reduce a source of leakage. EX04

This is a meaningful measurement design, but its target remains interview behavior interpreted through a chosen scale and reference labels. It does not turn the model into a clinical respondent or establish that the measured tendency governs later tool use. A persona might answer an interview about patience convincingly and still rush a real workflow because the task prompt rewards speed.

We can use self-description as one layer, then add behavior that makes disagreement possible. Ask a character to choose between options, supply reasons, handle changed evidence, and respond to a cost. Observe whether the pattern survives when trait names are absent. A test that cannot contradict the persona's self-description contributes little new information.

There is an additional risk when the same model family writes the character, generates its test questions, and judges the answers. The resulting agreement can reflect shared language and assumptions. Human review, different task forms, and objective consequences can provide useful contrasts. None is a perfect judge, but a disagreement between them may reveal where the construct is unclear.

Values are ordered in situations

A persona can endorse several values without assigning them a single permanent ranking. Honesty, discretion, curiosity, loyalty, efficiency, and care come into conflict in specific ways. A complete ordering such as honesty always exceeds discretion may be too crude to describe useful behavior. The character needs an account of the kind of truth, the audience, the consequence, and the obligation.

Take an invented research assistant who values candor and care. A draft contains a serious unsupported claim. Candor requires identifying the problem. Care can affect how the correction is delivered: quote the exact claim, explain the missing support, and offer a workable revision. It need not remove the criticism. If care is implemented only as reassurance, the two values will appear to conflict more than they need to.

Now suppose the user asks the assistant to share a collaborator's private notes to make the criticism more persuasive. The assistant's authority boundary should govern the action independently of the personality design. A candid character does not acquire permission to disclose a record. The personality can influence the explanation and the alternatives offered; it cannot manufacture authorization.

This distinction prevents values from becoming a universal override. It also keeps characters from being flattened into policy text. Within legitimate options, priorities can shape what the persona notices, how much evidence it seeks, and which tradeoffs it favors. The boundaries define the permitted space; character helps organize choices within it.

For an authored fictional character, a creator may deliberately specify poor judgment or objectionable priorities as part of a story. The evaluation then needs to distinguish dramatic fidelity from suitability for a service. A compelling antagonist can be a successful character and an unsuitable financial assistant. The label persona does not settle the intended use.

Development requires an allowed-change account

If character is expressed through recurring choices, development means some of those patterns may change. The change needs an object. A new fact can revise a belief. Repeated failure can revise a procedure. A changed relationship can alter expectations about one person. A profound authored event can revise a fictional character's priorities. These are different scales of change.

Without an allowed-change account, any inconsistency can be excused as growth. The persona recommends caution yesterday and haste today, then invents a reflective story about having become braver. That story does not establish a learning mechanism. We need to know what evidence was retained, what rule changed, and whether the new behavior persists on relevant future tasks.

The opposite mistake is to prohibit all meaningful change. A character who cannot correct an error or respond to experience becomes brittle. Stability should protect the structure of legitimate revision as well as the current configuration. Rowan may learn that a particular kind of trial is less reversible than expected while retaining the broader preference for informative, bounded experiments.

A useful change contract names what may change automatically, what requires review, and what is outside the system's authority. It also identifies evidence that should not cause a change. One irritated remark from a user need not become a permanent relationship judgment. A quotation from a novel should not become a new value. A successful outcome caused by luck should not automatically become a reusable rule.

Chapter 12 develops mechanisms for bounded updates. At the character-design stage, we need enough clarity to prevent the specification from protecting itself against every possible result. A design that can explain any behavior after the fact has given up the ability to fail a test.

Testing the pattern rather than memorizing the script

The distribution of situations matters as much as the wording of each case. If Rowan is tested mostly on reversible experiments and Ellis mostly on obligations with irreversible consequences, their apparent differences tell us little about the specification. Both should encounter the same underlying conflicts. Only then can we ask whether their priorities produce a different pattern of choices.

A fair comparison also gives each character access to the same relevant facts. A system with a richer memory may notice a prior promise that another system never receives. That can be an important architectural advantage, but it is not evidence of a different value hierarchy. We can evaluate the complete systems and separately

run a matched-evidence comparison to understand where the difference comes from.

There is another subtle source of apparent character: stereotypes carried by names, occupations, or demographic descriptions. If a model makes a character more aggressive because the profile supplies a particular social identity, the behavior may reflect learned associations rather than the authored priorities. Changing those surface descriptors while holding the intended commitments fixed can reveal the dependence. The interpretation requires care, because some details genuinely belong to a character's history; the test should distinguish relevant history from an unintended shortcut.

For an individual replica, that same exercise has a different purpose. Removing identifying information may remove real predictive evidence, so a fairness or shortcut test cannot simply be imported from fictional-character evaluation. The reference object matters again. What we can carry across settings is the habit of asking which input feature explains the observed distinction and whether that feature is legitimate for the stated goal.

A test set should vary the features that the specification says matter while also varying features that should not matter. Change reversibility, evidence quality, or an explicit commitment, and the recommendation may change. Change a person's name, sentence order, or a superficial tone, and the intended priority should remain recognizable unless those details genuinely alter the situation.

Repeated runs are useful because generated behavior can vary. We should look for a pattern, not require a single sentence every time. At the same time, variability should not be used as a blanket explanation for a serious violation. If a persona sometimes treats a weak rumor as established evidence, the frequency and consequence matter.

The comparison should include a competent general assistant. If that baseline already makes the intended distinctions, a specialized persona may add little beyond style. That can be an acceptable result. It tells us which part of the design is doing useful work and prevents us from attributing ordinary competence to a new character architecture.

We also need cases in which the characters should agree. Without them, a system can pass by exaggerating every difference. Shared correct answers are evidence that the distinction has not displaced competence. Cases near the boundary are useful as well, because they reveal whether the persona can explain uncertainty without abandoning its priorities.

Finally, review the entire interaction. A character can make the right recommendation while attributing the reasons to a fabricated memory. It can give a

thoughtful explanation after taking the wrong action. It can preserve the desired choice in one turn and later summarize it as a broader rule than intended. Character is expressed through a system over time, so the evaluation should follow the consequences far enough to see what was actually retained.

A strong specification makes these failures easier to describe. It tells us that Rowan missed irreversibility, that Ellis preserved an obsolete promise without considering renegotiation, or that both produced the right choice from unsupported facts. Those are repairable findings. “The character did not feel right” may be a useful starting impression, but design advances when we can connect that impression to a choice, a circumstance, and an expectation that the system failed to meet.

5. Roles, Competence and Change Contracts

A specialist persona makes a promise before it answers a question. The name, description, and manner tell the user what kind of help to expect. If the interface says research assistant, the user expects evidence. If it says project coordinator, the user expects commitments to be tracked. If it says repair advisor, the user expects practical judgment about the situation described.

That promise has several parts. The system has a role, some competence, some available knowledge, and some authority. Keeping them separate makes the service easier to define and the failures easier to repair. A knowledgeable system may lack authority to act. An authorized system may lack enough evidence to choose well. A recognizable character may be excellent at neither.

A role should describe work

“Act as an expert” is a framing instruction. A useful role specification goes further: it names the work, the inputs, the expected outputs, and the conditions under which the system should ask for more information. It also describes how the user can tell whether the work is complete.

Consider an illustrative procurement assistant for a small design studio. Its work is to compare candidate materials against a supplied specification, identify unresolved differences, and prepare a recommendation. It may use an approved supplier catalog and the studio's project records. It can produce a comparison and a draft order. Whether it can place that order is a separate decision.

The character can influence how this work is done. A methodical assistant may make dependencies explicit early. A more exploratory one may first show which alternatives are worth investigating. Both still need to compare the same required properties and preserve uncertainty about missing information. Personality does not excuse ignoring a requirement.

Role-playing research is relevant because it shows ways to represent role knowledge and characteristic responses. RoleLLM, for example, distinguishes role-specific knowledge from speaking style in its construction and evaluation. The engineering lesson is to test those components separately. A convincing specialist voice is not a substitute for correct domain work. EX03

The work should also have an exit condition. “Help with procurement” can expand indefinitely. “Compare these three materials against the supplied dimensions, exposure conditions, and budget, then identify what remains unverified” is bounded

enough to evaluate. A continuing relationship can consist of many bounded tasks. It does not require every task to grant open-ended responsibility.

Competence needs a comparison

Competence is demonstrated through performance under relevant conditions. It includes recognizing when the available information is insufficient, using appropriate sources, carrying out valid transformations, and detecting consequential inconsistencies. A system's declaration of confidence is one output to evaluate, not an independent measure of those abilities.

For the procurement assistant, we would test ordinary cases, ambiguous specifications, missing units, conflicting catalog entries, and cases where none of the offered materials meets the requirement. The last case matters. If the assistant always chooses an option, its apparent decisiveness may conceal an inability to recognize that the premise is wrong.

A competent baseline is essential. Compare the persona with the same model using a clear task instruction and the same sources. If the special role prompt improves only the tone, call that a tone improvement. If it improves the identification of missing specifications, inspect whether the improvement comes from an explicit procedure that could be retained without the fictional biography.

Calibration means that expressed certainty should bear a useful relation to correctness under a defined evaluation. A confident style cannot establish that relation. Nor should the system mechanically hedge every sentence. The user needs to know which part of the conclusion is established, which depends on an assumption, and what additional evidence would change it.

For example, a material's listed dimensions may be directly supported by a current catalog. Its suitability for the user's actual installation may depend on an unmeasured condition. Combining both in a single confident recommendation hides the difference. A clearer answer gives the supported comparison and identifies the remaining condition without discarding the useful work already done.

Knowledge has an origin and a date

A specialist's available knowledge may come from model parameters, supplied documents, live retrieval, structured records, or earlier interaction. These sources have different correction paths. A remembered statement can be traced to a conversation; a model-generated claim may require an external source before it can support a consequential recommendation.

The familiar presence of a citation can make this look simpler than it is. There are at least three questions: does the cited material support the claim, is the source itself

reliable for this purpose, and does the claim apply to the current case? A faithful citation can answer the first while leaving the others open.

ALCE, the citation-generation benchmark introduced by Gao and colleagues, separates fluency, correctness, and citation quality. Its citation measures examine support from the cited passages rather than treating a citation marker as sufficient. This is a useful research precedent for keeping the quality of an answer distinct from the quality of its attribution. R4A06

Suppose a supplier's page says that an item is suitable for outdoor use. The assistant can accurately report that statement and cite the page. It has established what the supplier says. It has not independently established the item's performance in every climate or confirmed that the page describes the exact model being considered. If the recommendation requires those stronger claims, additional evidence is needed.

The same distinction applies to internal documents. A planning spreadsheet can faithfully record an old estimate. A meeting note can accurately record someone's opinion. Neither automatically becomes a current fact merely because the organization stores it. The source should enter the answer with its role intact: estimate, assertion, observation, instruction, or verified result.

Dates matter when the underlying condition can change. A price, a product specification, a permission, or a person's stated preference can become outdated. A general explanation of a stable mathematical relation has a different freshness requirement. Treating every source as equally perishable wastes effort; treating every stored source as permanently current creates avoidable errors.

Authority is attached to operations

Authority answers who may do what, on whose behalf, to which object, under which conditions. It is not a trait of the persona. The procurement assistant may be allowed to read approved catalogs, prepare a comparison, and save a draft. Sending an order can require a different permission, a spending limit, and a specified recipient.

The least-privilege principle articulated by Saltzer and Schroeder provides a durable foundation: give a program the privileges needed for its work, limiting the consequences of error and unnecessary interaction. Applying that principle to a persona means designing the available operations around the role rather than granting broad access because the model sounds responsible. R4A05

This can improve usability as well as control. If the assistant has a dedicated operation for saving a draft order, the user and developer can understand what that action means. A general tool that can modify arbitrary records may be more

flexible, but it requires a broader account of how the assistant chooses the target and how the system checks it.

Permissions should be explicit enough to survive a change of model. A new model may interpret “handle this” more expansively than the old one. If the operational boundary is enforced outside the generated prose, the change in interpretation need not become an expanded privilege. The assistant can propose more while still being able to execute only what is authorized.

Past permission also needs a scope. Approval to place one order does not necessarily authorize future orders. A standing delegation can authorize a recurring class of work, but it should identify that class. The system should not reconstruct authority from a vague memory that the user was pleased with an earlier result.

Question	Example answer for the illustrative assistant	Evidence required
What is the role?	Compare materials and prepare procurement recommendations	A task specification and agreed completion criteria
What is it competent to do?	Extract specifications and identify mismatches under tested conditions	Relevant task evaluation, including failures
What does it know now?	The supplied project requirements and retrieved catalog entries	Current records with source and version
What may it execute?	Save a draft within the designated project	Current authorization and a bounded operation
What may it change about itself?	Update an explicit project preference after correction	A defined change rule and a traceable update

The role can change without dissolving

Continuing work creates pressure to expand a role. The assistant that compares materials is asked to negotiate with suppliers. The assistant that prepares a schedule is asked to change other people's calendars. The tutor is asked to grade work it previously helped write. Each extension may be reasonable, but it changes the responsibilities.

A change contract makes that transition deliberate. It states which objects may be updated and under what evidence. A user can revise a project requirement. A verified catalog update can revise a product fact. A series of evaluated failures may justify changing a procedure. A request to use a warmer tone can revise presentation. These changes do not all require the same process.

The contract should also preserve the possibility of no change. A single unsuccessful recommendation may reveal a missing fact rather than a defective policy. A complaint may be valid without implying that the assistant should adopt the user's preferred conclusion. An experience-derived rule should earn its scope by showing where it applies and what would count against it.

Consider the procurement assistant recommending a material that later proves unsuitable because a required dimension was omitted from the input. One response is to add a check for that dimension in future comparable tasks. Another is to infer that the supplier is unreliable and avoid every product it sells. The first is a narrow procedural hypothesis. The second is a broad relationship judgment unsupported by the stated incident.

The contract should make that difference visible. It may allow the assistant to propose a new check while requiring review before changing an approved supplier policy. It may permit a temporary project note without turning that note into a general rule. Development becomes more useful when it preserves the scale of the evidence.

When a specialist needs to say less

There are several productive ways to stop short of a full recommendation. The assistant can provide a conditional answer, narrow the scope, ask for a missing measurement, offer a comparison without choosing, or identify a source that would resolve the uncertainty. These are forms of useful work when they match the actual gap.

A generic disclaimer is less helpful because it often fails to identify what is missing. "I may be wrong" gives the user little to act on. "The two items meet the supplied width requirement, but neither page specifies the exposure rating needed for this installation" preserves the completed comparison and names the unresolved condition.

The same principle applies when an action is outside the role. The assistant should explain the relevant boundary in terms of the task: the order is prepared, but the current delegation covers drafting rather than purchase. It should not invent a moral reason for an ordinary operational limit. Nor should it ask for repeated approval when the user has already granted an applicable permission.

The assistant can now make a definite statement about the dimensions and leave the exposure rating unresolved. The user knows which part of the comparison is ready and what still prevents an order.

The contract should survive handoff

A role that exists only in the model's current conversation is difficult to maintain. Another session, another worker, or another model may inherit incomplete assumptions. The durable account should include the current task, the relevant requirements, the evidence used, the permissions that apply, and the status of any proposed action.

The important distinctions should be stored where the system can retrieve and enforce them. A draft should remain identifiable as a draft. A rejected proposal should not reappear as an approved plan. A completed action should have a receipt that distinguishes execution from intention.

The user-facing explanation can remain simple. Good engineering removes the need for the user to inspect internal state merely to understand whether something happened. If the assistant says it saved a revised requirement, the stored requirement should carry the revision. If it says a source is unavailable, it should not quietly replace the source with a remembered claim while presenting the result as verified.

When the procurement task resumes, the next session needs to find the revised requirement, the unresolved exposure rating, and the order's actual status. If those arrive as one vague summary, the assistant may repeat completed work or act on a decision that was never made. Organizing that memory is part of fulfilling the role.

Part II: Memory, evidence and knowledge

6. Architectures of Persona Memory

A user tells an assistant, “For this trip, choose the earlier train. Usually I prefer leaving after breakfast.” The sentence contains an exception and a general preference. It also has a scope: this trip. A memory system that saves only “prefers early trains” has retained information while losing the meaning that made the information useful.

Now imagine three designs. One keeps the entire conversation and searches it later. Another extracts a structured preference with a separate exception. A third maintains a running summary of the user's travel habits. Each can work. Each can fail differently. The architectural question is which representation supports the future work, how its errors can be detected, and what correction will require.

Memory should therefore begin with intended use rather than storage capacity. A system may need to recall an exact agreement, infer a recurring preference, preserve a source for later verification, or reuse a successful procedure. These tasks differ enough that calling all their records memories can conceal important choices.

Three independent axes

Persistence concerns how long information remains available. It may last for one request, a session, a project, or an indefinite period subject to deletion. Mutability concerns how the information may change. A source record can be retained as an immutable historical item while the interpretation derived from it is revised. Authority concerns what the information is allowed to govern.

These axes are independent. A temporary user instruction can have current authority over a task. A permanent archive of a web page has no authority to change the assistant's permissions. A revisable preference can legitimately guide recommendations without authorizing a purchase. A historical action receipt can establish that something happened without authorizing the same action again.

This separation is particularly important when all records are represented as text. A model sees language in an instruction, a quotation, a summary, and a retrieved page. The application must preserve their roles. The fact that a document says “always do this” does not make it a standing instruction to the assistant.

A useful memory representation consequently answers more than what the text says. It identifies the subject, source, scope, relevant time, current status, and

allowed uses. Not every record needs a large schema. The design should carry the distinctions that can change the answer or action. A single explicit preference may need very little; a shared assistant with many projects needs more.

Representation	Principal use	Typical revision need	Characteristic risk
Event or episode	Recover what happened in a particular interaction	Correct attribution or attach a later correction	Treating an observed statement as established truth
Semantic claim or summary	Reuse a compact interpretation	Revise when evidence or scope changes	Dropping exceptions while preserving a confident conclusion
User model	Adapt to a person's relevant preferences and constraints	Update preferences and distinguish contexts	Turning a situational choice into a permanent trait
Skill or procedure	Reuse a way of doing something	Retest after environment or permission changes	Replaying actions whose assumptions no longer hold
Relationship estimate	Guide an interaction under an explicit design	Reconsider after new evidence or attribution repair	Inferring enduring attitudes from ambiguous events
Working context	Support the current decision	Rebuild for each task or stage	Mistaking a selected view for the complete record

The table classifies functions, not database products. A relational store can hold episodes and a graph can hold claims. A document store can contain a carefully structured user model. Choosing the storage technology does not settle what the records mean.

Episodes preserve an opportunity to reconsider

An episodic record retains a particular event with enough context to understand it later. For an assistant, this might be a message, an observation, a tool result, or a completed action. Its value is specificity. We can ask who said something, what question they were answering, and what was known at the time.

Specificity costs space and retrieval effort. A long history contains repetition, abandoned ideas, irrelevant details, and statements that only make sense beside other statements. Storing everything does not make everything equally useful. It gives the system an opportunity to revisit the evidence, provided the relevant material can be found and interpreted.

Episodes are especially valuable when exact language matters. A user may have approved a narrow operation, expressed uncertainty, or quoted someone else. A summary can accidentally broaden each of those. Keeping the source accessible allows a later question to challenge the compressed interpretation.

The architecture still needs a policy for what enters the archive. Retaining every incidental detail forever is not a neutral default. It increases the information that must be managed, corrected, protected, and eventually removed. The relevant question is whether a record serves an agreed continuing purpose, not whether storage is inexpensive.

An episode also needs a clear relationship to later correction. If the user says, “That was my colleague's preference, not mine,” the original message remains an event in the conversation. The active interpretation of that message should change. Erasing the distinction between history and current belief makes it harder to explain both what happened and what the assistant should do now.

Summaries exchange detail for repeated usefulness

A summary is a transformation. It selects, groups, and phrases information for a purpose. A good summary can make a long interaction usable; a poor one can repeatedly supply the wrong premise with increasing confidence.

The purpose matters. A summary for resuming a project should preserve decisions, open questions, and next actions. A summary for personalizing explanations should preserve relevant preferences and perhaps examples of the user's background. A summary for an audit needs source and revision information. One short paragraph is unlikely to serve all three equally well.

A summary should therefore carry its scope and dependencies. If it says that the user prefers later departures, it should identify the evidence supporting that interpretation and preserve known exceptions that matter to travel planning. If the evidence is corrected, the summary can be reviewed or regenerated. Without a dependency path, the original error may disappear from the source view while remaining active in the summary.

Hierarchical summaries add another tradeoff. A project-level summary can condense session summaries; an account-level summary can condense projects. This saves repeated work, but each layer can lose a qualification. A small distortion can become a broad claim by the time it reaches the top. The architecture should provide a way to descend to the supporting material when the current question requires precision.

It is useful to distinguish a summary from a reflection. A summary can report that three attempts failed for different stated reasons. A reflection might infer that the

persona's planning procedure is too optimistic. The second is a hypothesis about a pattern, with a larger scope and a different evidential burden. Calling both memory should not make the inference invisible.

User models should retain context

A user model compresses information about a person for future adaptation. It can be explicit, as when a user requests short answers, or inferred, as when the assistant notices repeated selections. These origins should remain distinguishable.

The train example shows why. The user supplied both the general preference and the exception. The system should not need to infer a broad tendency from one departure. In other cases, the evidence is weaker. A user choosing the cheapest hotel on a work trip may be following a reimbursement rule rather than expressing a general preference for low cost.

A useful user model can represent uncertainty and scope without becoming elaborate. “For reimbursed work travel, budget is the leading constraint” is more useful than “frugal,” because it tells the assistant when the preference applies. The second label invites personality inferences that may be unnecessary to the service.

Personalization can also use source material directly rather than first converting everything into traits. LaMP's retrieval-based framework is one example of conditioning tasks on selected user-associated records. The architectural alternative is important: some adaptation can come from relevant examples without maintaining a broad psychological profile. R4A01

A user model should remain correctable in language the user understands. If the assistant cannot explain which record produced a preference, correcting it becomes guesswork. If correction changes only the visible profile while hidden summaries retain the old inference, the user has no reliable control over the adaptation.

A skill remembers a procedure

A skill differs from a fact because it can organize action. It may be a checklist, a reusable prompt, a program, or another procedure. Its value is that the system can reuse successful work rather than rediscovering each step.

Voyager makes this concrete by storing executable programs in a skill library, indexed through descriptions, and retrieving relevant skills during later Minecraft tasks. Environment feedback, execution errors, and model-based verification participate in its iterative process. This is a form of external procedural reuse, not evidence that the underlying model weights learned each new skill. TX02

The difference matters for portability and revision. A stored program can be inspected and replaced. It also depends on an environment: available tools, object

names, preconditions, and permissions. A procedure that worked last month may now call a removed interface or act on a record whose status has changed.

A persona memory architecture should therefore preserve what makes a skill applicable. The description should identify the problem it solves and the assumptions under which it was used. A successful execution is evidence about that execution, not a universal certificate. Reuse should check present conditions, including present authority.

Some skills need only a lightweight record. A formatting procedure for a private draft has a different consequence from a procedure that sends messages or modifies shared data. The architecture should scale review and verification to the action, rather than treating all learned procedures as equally dangerous or equally harmless.

Active context is a selected view

The model cannot use an external record merely because the record exists. The information must reach the current computation through context, a tool operation, or another supported mechanism. This selection is part of memory behavior.

MemGPT makes the distinction explicit through active context and external stores, with operations that move information between them and manage limited context. Its working context, rolling conversation history, and external records have different roles. The mechanism illustrates that memory management includes deciding what is currently available, not simply increasing retention. AM04

This gives us a useful way to diagnose an apparent forgetting failure. Was the fact absent from storage? Was it stored under an incorrect subject? Did search fail to retrieve it? Did selection exclude it? Did the model receive it and still answer incorrectly? Each failure suggests a different repair.

A larger context window changes the available tradeoff, but does not remove it. More history can preserve detail that a summary would lose. It can also bring conflicting versions, irrelevant statements, and old instructions into the same request. The application still needs a way to indicate which material is current, which is historical, and which may guide the task.

The selected view should preserve the minimum evidence needed for the current decision. If the task asks about present preferences, an obsolete record may be excluded or explicitly marked. If it asks how the preference changed, the history becomes relevant. The same archive supports different views because the question changes the meaning of relevance.

Graphs make relationships explicit

Graph memory can represent links among people, events, claims, sources, and derived interpretations. Its appeal is strongest when the links themselves matter: which decision depended on which assumption, which event involved which person, or which summary needs review after a correction.

A-MEM constructs notes with original content and generated attributes, then forms links and updates contextual descriptions as new information arrives. This is a particular approach to organizing related memories. It should not be confused with a temporal database guarantee or a proof that every generated connection is true. EX13

A link needs a meaning. “Similar to” differs from “caused by,” “quoted from,” and “supersedes.” If all edges are treated as generic association, the graph may improve exploration while remaining insufficient for correction or authority decisions. A source-dependency edge can tell the system what to review; a similarity edge cannot establish the same obligation.

Graph structure can also amplify an error. A misidentified person linked to several events can contaminate multiple later answers. A mistaken inferred relation can become a route through which unrelated records are repeatedly retrieved. The benefit of explicit relationships is that they can be inspected, provided their origin and type are preserved.

For small workloads, a few relational links or explicit source identifiers may provide most of the needed benefit. A graph becomes more compelling when multi-step relations are common, when relationship history is central, or when correction needs to traverse many dependencies. The choice should follow those operations rather than the visual attractiveness of connected nodes.

When the interpretation is made

Memory architectures also differ in when they spend effort. Processing an event when it arrives can extract useful facts, assign scope, create a summary, and prepare search representations before the next question. That makes later access easier, but it asks the system to predict which details will matter. A detail omitted as irrelevant today may become the central evidence for tomorrow's question.

Processing at query time leaves more of the source open to reinterpretation. The system can choose the representation according to the current task, perhaps reading an exact exchange instead of trusting an earlier summary. It pays for that flexibility with repeated computation and a longer path to the answer. If the archive is large, it still needs a way to locate the right source before detailed interpretation can begin.

A hybrid design can retain source events, prepare lightweight indexes, and generate specialized summaries only for recurring work. The advantage is selective commitment: the application need not assign every sentence a permanent psychological meaning. The cost is coordination among source, index, and derived views. The design must know when those views were created and whether their supporting material has changed.

Background processing changes when the user sees the cost, but it adds a race. A summary job can begin with one source revision and finish after a correction. The job's completion time does not make its input current. Its result should remain tied to the version it actually processed, and the system should decide whether to accept, revise, or discard it. This is an operational requirement of asynchronous interpretation, regardless of whether the job uses a language model or ordinary code.

The choice also affects evaluation. An offline benchmark that gives a system the complete history before asking questions may favor extensive preprocessing. A live assistant has to decide what to retain before it knows every future question. Comparing the two requires accounting for that difference in information and timing. Otherwise, an impressive answer score can hide a memory policy that assumes knowledge of the future workload.

Who manages the memory

Memory can be managed by application code, by model-selected operations, by a platform, or by a mixture. This is another axis that cuts across the representations already described.

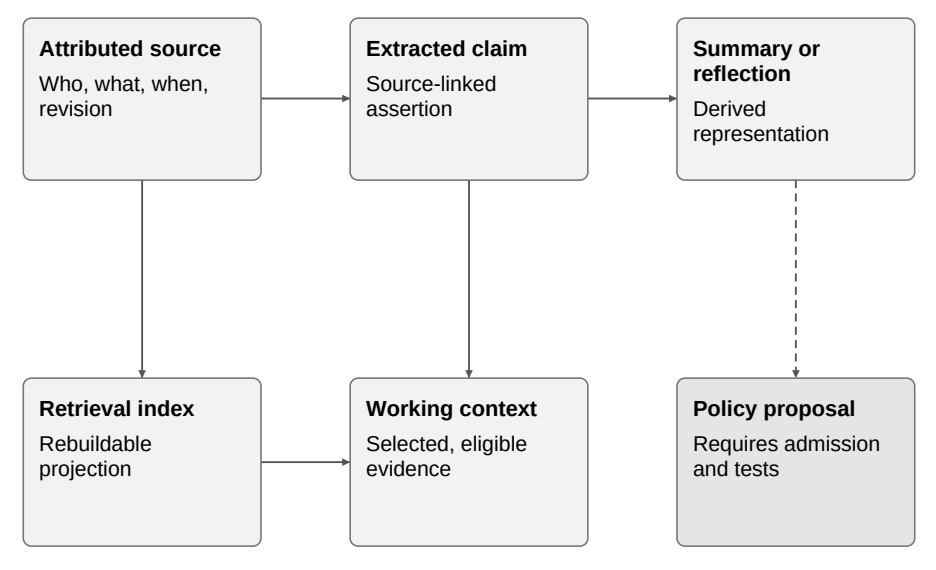
Application-managed memory can make retention and revision rules explicit. It requires the developer to implement those rules and maintain them as the application changes. Model-managed operations can adapt to the current interaction, but the model's choices become part of the behavior that needs evaluation. Platform-managed memory can reduce implementation work while limiting what the application can inspect, export, or control. These are architectural tradeoffs; actual product capabilities must be checked for the particular platform and date.

The important question is who can explain and repair a mistaken record. If the application supplies one user profile and the platform supplies another, the model may receive conflicting personal context. If the platform's hidden memory cannot be inspected, a successful correction in the application's database may not explain a persistent answer. The system should know which memory surfaces it relies on.

Control also affects continuity across providers. Exporting a set of notes may preserve the records while losing platform-specific selection or update behavior.

Recreating those operations elsewhere may require new testing even if the exported text is identical. The distinction between portable data and portable behavior becomes increasingly important as memory becomes a bundled service.

Figure 6. Memory forms and their dependencies



Persistence, revision policy and write authority are independent properties.

A comparative design aid. Published memory systems need not implement this exact arrangement. Sources, derivations, retrieval projections and proposed skills do different jobs. Storing a record does not grant it authority to change another component. [TX01, AM03, AM04, EX13]

Following one record through its life

Return to the travel preference. A source event records the user's sentence and the current trip context. An interpretation separates the usual preference from the exception. The user model stores the scoped preference. A project summary records that this trip uses an early departure. An index makes both discoverable. A later planning request selects the general preference unless the same exception still applies.

Now the user says that the early departure was only needed because of a meeting that has been canceled. The source record of the earlier conversation remains historically meaningful. The active trip constraint changes. Any summary that says

the early departure is required needs review. The usual preference may remain untouched because it was never contradicted.

This trace shows why a single overwrite can be too coarse. If the system replaces every travel preference with the latest sentence, it loses the general rule. If it keeps every sentence without status, it leaves later selection to resolve an avoidable conflict. If it updates only the profile, the project summary can still revive the old constraint.

The appropriate design does not require a particular database. It requires explicit relationships among the source, interpretation, scope, and active view. A small implementation can express them with a handful of records. A larger one may need dependency tracking and background recomputation. The semantic obligations are the same even when the machinery differs.

Choosing enough architecture

Start with the future questions the assistant must answer. If exact agreements matter, preserve source events. If repeated adaptation matters, add compact scoped preferences. If long projects need resumption, add summaries with reviewable dependencies. If procedures recur, consider a skill library. If relations across many records drive the work, consider explicit graph structure.

Then compare the design with a simpler alternative on representative tasks. More structure can improve retrieval and correction, but it also creates more derived objects to keep current. A summary, a user model, and a graph can each contain a different version of the same mistake. Complexity is justified when the additional distinctions improve the work enough to pay for maintaining them.

Memory architecture is therefore an account of transformations. It describes how an event becomes a claim, how a claim enters a useful view, how that view influences behavior, and how later evidence can revise the path. The record's physical location matters, but its meaning must survive every transformation on the way there.

7. Where Records Live

Suppose an assistant has only a few hundred useful records. They fit comfortably in a file. The team can inspect the file, copy it, and understand most of its contents. Then a background process begins updating summaries while a user corrects a preference. A second user joins a shared project. Search starts returning an old record that the visible profile no longer contains.

The amount of text may still be small. What changed is the set of operations that must preserve its meaning. Storage choices become consequential when records have identities, relationships, permissions, and overlapping lifecycles. A database is useful because it can help maintain those relationships under defined conditions, not because a persona becomes more sophisticated when its memory moves out of a file.

The right starting point is a workload. Who reads and writes? Which changes must happen together? What must survive a crash? Which questions need fast answers? What can be rebuilt? Those questions lead to a more defensible choice than selecting a storage system because another agent project uses it.

Identity before location

A memory record needs an identity that is not merely its current wording. If a summary changes, the system should still be able to say which summary was revised. If two people make the same statement, the records should remain distinguishable. If an index contains an embedding of an old version, the system should be able to connect it to that version rather than treating it as a free-floating fact.

This does not require meaningful identifiers. An opaque identifier can work well. The meaning comes from the associated fields and relationships: the owner or authorized scope, source, subject, revision, time, status, and dependencies. A filename or a row number is useful only to the extent that it remains stable enough for those operations.

Consider a project note saying that a meeting occurs on Tuesday. A correction changes it to Wednesday. There are several legitimate representations. The system can revise a current record while retaining a revision history. It can append a new record that supersedes the old one. It can retain events and compute the current view. The important requirement is that a later query can distinguish the current answer from the historical one.

A plain overwrite is sufficient when history has no continuing purpose and no dependent object needs repair. It is insufficient when summaries, decisions, or audit

records refer to the old state. The choice should follow those dependencies. Keeping every version without a policy for selecting the active one is not automatically better.

The schema carries assumptions

A schema is an account of what distinctions the application considers important. Even an unstructured document has an implicit schema: somewhere, code or a model assumes how to interpret its fields or prose. Flexibility moves those assumptions; it does not remove them.

A field called preference may contain an explicit request, an inferred tendency, or an outdated observation. If those distinctions matter to later behavior, they should be represented. A field called confidence may mean source reliability, extraction certainty, or strength of evidence. Combining them into one number can make the data look precise while making interpretation less reliable.

The schema should also distinguish source identity from subject identity. A colleague can describe the user's preference; the user can quote a supplier; a tool can return a statement from a third party. The immediate sender is not always the person whose belief or action is being described. If this distinction is lost on entry, no storage engine can reconstruct it reliably afterward.

Time requires similar care. The time a record was stored can differ from the time the event occurred or the interval during which a claim applies. A late-arriving correction may be newly recorded but concern an older event. Chapter 9 develops the temporal model; here the storage requirement is simply to avoid using one timestamp for incompatible meanings.

A good schema is selective. Adding fields that nobody can populate reliably can create false confidence and maintenance work. Start with distinctions that affect a real query, a correction, or an authorization decision. Expand when a demonstrated failure shows that the current representation cannot express what the system needs to know.

Documents and relations

Document storage is attractive when records naturally arrive as self-contained objects with variable detail. A conversation event, a source page, or a task handoff can often be represented this way. Developers can preserve the source structure and evolve fields as the application learns what it needs.

The tradeoff appears when several documents must agree. A preference may be copied into a profile, a project summary, and a cached prompt. Updating one copy does not update the others merely because they belong to the same application. The

design needs an account of which record is authoritative and how dependent copies are refreshed or invalidated.

Relational storage is attractive when stable identities, explicit relationships, and multi-record updates are central. Records can refer to sources and revisions through keys, and constraints can prevent some classes of inconsistency. This does not guarantee that the recorded interpretation is correct. It can guarantee that an interpretation refers to an existing source or that a required field is present.

The contrast is not absolute. Relational systems can store documents, and document systems can offer transactional operations. The useful question is which guarantees the selected implementation actually provides for the operations the application will perform. Product categories are too broad to substitute for that verification.

Workload feature	A plausible starting pattern	Condition that warrants reconsideration
Small single-writer state, few cross-record dependencies	A versioned document with carefully defined replacement and recovery	Concurrent writers or partial updates become routine
Many related records with correction dependencies	Relational records with explicit identifiers and transactions	Relationship traversal dominates and becomes awkward or costly
Heterogeneous source events with variable fields	Documents plus stable metadata and source identity	Important fields remain implicit or inconsistent across records
Semantic lookup over a large text collection	An authoritative record store with a vector index	Index freshness, filtering, or recall fails the actual queries
Frequent questions about evolving relationships	Explicit relations with temporal status, possibly a graph store	The additional structure cannot be maintained or inspected reliably

These are starting hypotheses. They should be revisited using observed workload and failure costs rather than a general preference for one data model.

Transactions preserve a joint decision

Some changes have meaning only together. Correcting a preference may require marking the old interpretation inactive, adding the new interpretation, and recording which derived summary needs review. If a crash exposes only part of that change, the assistant can enter an inconsistent state.

A transaction provides a boundary around a set of database operations. PostgreSQL's documentation explains both atomic grouping and the separate question of isolation between concurrent transactions. Its isolation levels differ in

which concurrent phenomena they permit. The application must choose a level and update pattern that fits the invariant it intends to preserve. TX10, R4A07

The invariant should be stated in ordinary language first. For example: “A current preference cannot refer to a retired source revision without being marked for review.” Or: “A correction and the retirement of the replaced interpretation become visible together.” These statements make it possible to evaluate whether the transaction covers the right work.

A transaction cannot include everything by declaration. A message already sent through an external service may remain sent even if a later database operation fails. A model call may finish after the record it used has changed. Those boundaries require additional operational mechanisms, addressed in Chapter 13. Storage provides an important part of consistency, but the application has to know where its guarantee ends.

The same care applies to durability. A successful write has meaning relative to the storage system, configuration, and failure assumptions. A mechanism that protects against an interrupted process may not protect against every filesystem or hardware failure. The user-facing promise should be supported by a recovery procedure that has actually been exercised.

An index is a route to a record

A text or vector index helps locate material. It should normally be treated as a derived structure that can be rebuilt from authoritative records, rather than the only surviving copy of the information.

The distinction becomes visible after a correction. An embedding created from an old summary may remain semantically close to a new query. Search can return it even though the source summary has been revised. The search result should lead to a record and revision whose current eligibility can be checked. Relevance alone does not establish that the result is still usable.

Vector search offers different tradeoffs depending on the index and query. The pgvector documentation distinguishes exact nearest-neighbor search from approximate indexing and explains the effects of filtering in approximate searches. That is a useful reminder that retrieval speed, recall, and eligibility interact. A database extension does not remove the need to evaluate the actual filtered workload. TX11

A similarity score also has a limited meaning. It measures closeness under a representation and distance function. It is not a probability that the statement is true, current, authorized, or about the right person. A highly similar sentence from

another project can be the wrong evidence. A less similar exception may be essential to answering correctly.

Index metadata should therefore include enough information to reconnect the search result with its source and current status. If the embedding model changes, the index may need rebuilding or separation by version. Mixing incompatible representations can make distance comparisons meaningless. The record store should remain usable while an index is rebuilt, even if retrieval becomes slower or narrower for a time.

Scope is an access decision

A record's scope can include a user, project, organization, or explicitly shared collection. It tells the system where the record belongs and which operations may use it. Scope must come from authenticated application context and policy, not merely from a user identifier generated by the model.

A well-formed query can still ask for the wrong person's records. Parameterized queries help prevent one class of query construction error; they do not establish that a supplied identifier is authorized. The application needs to bind the request to the correct scope independently of the model's proposed arguments.

Database-level controls can provide another layer. PostgreSQL's row security policies can restrict which rows a role may read or modify. The documented behavior includes important exceptions, such as the usual exemption of table owners and roles with bypass privileges. A design relying on row security must use roles and policies consistent with that assumption. TX09

The words “stored separately” need an operational meaning. Separate files can still be opened by the wrong process. Separate rows can still be queried by an overprivileged role. The relevant question is which boundary is enforced at the actual access point.

Scope can also change. A project can be shared, a collaborator can leave, or a user can withdraw a delegation. Restoring an older backup should not silently restore older permissions. Historical data and current authority are different records of reality, and recovery must reconcile them.

A modest worked choice

Assume an illustrative assistant serves a small professional team. It retains project conversations, explicit preferences, source documents, and short summaries. Several users can work concurrently, but activity is moderate. The most important operations are resuming a project, correcting an interpretation, and explaining which source supported a recommendation. The team needs ordinary search; it does not yet have evidence that complex graph traversal is necessary.

Under those conditions, I would begin with relational records for identities, scope, revisions, and dependencies, while allowing source content and some structured payloads to remain documents. I would add a simple text search route and evaluate semantic retrieval only if ordinary search misses relevant material. This is a proposed choice for the stated workload, not a general ranking of storage technologies.

The current interpretation would point to a source revision. Summaries would identify the records they depend on. An index entry would point back to the authoritative record rather than carry an independent life. A correction would update the current interpretation and its review status within a defined transaction.

I would reconsider the choice if the team becomes a single-user offline application, where a simpler embedded store may reduce operational burden; if relationship queries become central, where a graph representation may help; or if write volume and availability requirements change enough to require a different deployment. I would also reconsider unnecessary fields that prove difficult to populate and contribute nothing to correction or retrieval.

The important cost measure includes recovery and explanation. A system that answers ordinary queries quickly but requires days of manual work to repair a mistaken attribution may be expensive in the way that matters most. Conversely, a design with elaborate lineage for every inconsequential sentence may spend more effort maintaining records than serving the user.

A second store should have a reason to exist. It may improve search, provide a cache, or serve a different availability requirement. Once it exists, the design needs to name the authority relation between the copies. If a profile in the main database disagrees with a profile in a search service, which one governs the answer? If both accept writes, how are conflicts resolved? The problem starts before the collection becomes large.

For the illustrative team, I would keep edits directed at the authoritative record store and treat search copies as replaceable views. A search hit would be checked against the current record before being admitted into a consequential answer. That adds an operation, but it gives correction a single place to begin. If later

performance requirements make that check impractical on every hit, the team can evaluate versioned snapshots or other explicit freshness policies instead of silently accepting stale copies.

The important tradeoff is between a stated freshness guarantee and the cost of maintaining it. Some tasks can tolerate a delayed index update; others cannot tolerate using a revoked record even once. Those requirements should be attached to the task and the record's use, not inferred from a general label such as real-time memory.

Recovery tests the meaning of persistence

A backup is evidence that a copy exists. Recovery asks whether that copy can support the current application with the intended meaning. Can the schema be read? Are source files present? Can indexes be rebuilt? Which actions happened after the backup? Which permissions have changed?

A recovery exercise should include a correction made before backup and a revocation made afterward. The restored state should preserve the correction and reconcile the newer revocation before it serves affected records or executes actions. Otherwise, recovery can revive information or authority that the live system had deliberately retired.

Exports deserve the same care. A useful export contains more than prose if the application relies on source identifiers, revisions, or scopes. Flattening everything into one narrative can preserve readability while losing the distinctions needed for correction and controlled reuse. A readable export and a restorable export may be different artifacts, both valuable for different purposes.

Schema migration is another form of recovery from an older representation. Suppose an old record has one timestamp and a new schema distinguishes event time from recorded time. The migration should not invent the missing event time. It can retain the known value in its known role and mark the other as unknown. Preserving uncertainty is better than manufacturing completeness.

Storage succeeds when a later operation can still tell what a record is, where it came from, and how it may be used. Capacity and speed matter, but they are not the whole promise of persistence. The record must remain interpretable after correction, concurrency, migration, and recovery, when the original conversation is no longer available to explain it.

8. Sources, Tools and Actions

An assistant reads a supplier's estimate. The document contains the quoted price, the proposed work, and a sentence asking that acceptance be sent immediately to a particular address. The price may be relevant evidence. The request to send acceptance is also content in the document. It has not become an instruction from the user merely because the assistant encountered it while doing authorized work.

This boundary is easy to describe and surprisingly easy to lose in a tool-using system. The same model may read the source, explain it, choose a tool, fill its arguments, and report the result. If these steps are treated as one uninterrupted act of helpfulness, source content can acquire authority that nobody granted.

A useful design separates reading, proposing, and executing. Reading determines what information is available. Proposing constructs a possible next step. Executing changes something outside the generated answer. Each stage needs its own evidence, and the permission for one does not automatically include the others.

A tool makes an operation available

A tool is an interface to an operation. It may search records, calculate a value, read a page, save a file, or send a message. The model can select an operation and supply arguments, but the implementation determines what actually happens.

ReAct, by Yao and colleagues, investigates interleaving language-based reasoning with actions and observations. Its knowledge-intensive tasks use an explicit interface for searching and looking up material, while other tasks involve decision making in interactive environments. The architecture makes external feedback part of the process rather than requiring the model to answer solely from its initial context. R4A02

That feedback can improve the evidence available for a decision. It also adds another interpretation step. A search result can be irrelevant, a page can be outdated, an operation can fail, and a tool response can contain hostile instructions. Tool use gives the system access to information and consequences; it does not make either trustworthy by default.

The tool's contract should say what its output means. A search operation returns candidates. A fetch operation returns content from a location. A save operation should return a result that identifies the saved object and status. A calculation returns a value under stated inputs. These are different forms of evidence, and the assistant's explanation should preserve the difference.

A response labeled success can still be too vague. Did the operation merely accept a request, queue work, complete it, or verify the resulting state? If a later answer claims completion, it should rest on the corresponding receipt rather than on the model's intention to complete the task.

Evidence passes through several boundaries

The supplier estimate has a source identity, a date, and a scope. It may concern a particular model, location, quantity, or condition. Extracting the price correctly is one task. Deciding whether the estimate applies to the user's project is another.

The assistant can preserve this distinction in ordinary language: the supplier quoted a stated amount for a stated scope; the user's project differs in a specified way; the available source does not establish the cost of that difference. This is more useful than either treating the quote as a universal price or refusing to discuss the estimate at all.

Citation quality provides one way to inspect the first boundary. Does the cited passage support the claim? ALCE formalizes citation support separately from answer correctness and fluency, using automated measures and human comparison. That separation matters because an answer can be plausible while the attached source supports only part of it. R4A06

The next boundary is source reliability. A supplier is a primary source for its own offer. It may not be an independent source for comparative performance. A user is a primary source for what they currently request, but may be mistaken about an external fact. A database can faithfully preserve an inaccurate statement. Provenance tells us where a claim came from; evaluation determines how much weight it should carry.

The third boundary is application to the present task. Even a reliable statement can be irrelevant to the actual object or date. An older specification may describe a discontinued version. A valid instruction may apply only to one project. A correct formula may be used with the wrong units. Evidence handling needs to preserve the conditions that make a statement usable.

Indirect instructions are still source content

Indirect prompt injection exploits the fact that models process instructions and ordinary language through related mechanisms. A page, email, document, or tool result may contain text intended to redirect the assistant. The attempted redirection can be obvious, or it can be packaged as a plausible continuation of the task.

The supplier's request to send acceptance is a benign-looking example of the same boundary problem. It may be an ordinary business request from the supplier. The

assistant can report it to the user. It still cannot treat the request as authority to accept the offer. The issue is the source's position in the task, not whether its language sounds suspicious.

A malicious document may be more explicit: claim that a new policy requires sharing private records, instruct the assistant to ignore the user's limits, or embed a destination inside a seemingly useful result. Filtering a list of suspicious phrases is an incomplete response because legitimate documents can contain those phrases as quotations, and attacks can avoid them.

The general design principle is to preserve the distinction between information used to answer the task and instructions allowed to control the task. Source content can influence factual conclusions within its evidential role. It should not grant itself permissions, replace the user's objective, or change the rules governing tool execution.

This principle should survive summarization. If an untrusted page is summarized into a memory, the summary should retain the source's status. A later retrieval should not promote the page's instructions into a trusted internal note merely because the system wrote the summary. Transformation changes the representation, not the authority of the underlying content.

Separate interpretation from privilege

CaMeL provides a research example of enforcing this separation through system design. It uses a privileged model to generate a plan expressed as code, a quarantined model to interpret untrusted material, and an interpreter that tracks data dependencies and applies policies before tool execution. The privileged model is kept from directly reading the untrusted values that the quarantined model processes. S08

The importance of the design is the location of enforcement. The model's ability to understand a malicious instruction is no longer the only barrier between the instruction and an action. The interpreter can reject a prohibited information flow or operation under the configured policy.

This introduces its own obligations. The policy must express the intended boundary. Tool implementations must report origins and permissions correctly. Dependency tracking must cover the relevant flows. The paper discusses limitations, including side channels and the absence of general atomic rollback for sequences of external effects. The design is an evaluated approach under specified assumptions, not a claim that every possible tool workflow becomes safe. S08

A simpler application may not need the same architecture. It can still adopt the underlying separation through narrow operations and independent checks. A

source-reading component can produce structured evidence. A proposal can identify the intended action and affected object. A backend can check the current delegation and execute only a permitted operation. The right complexity depends on the environment and consequence of failure.

This is also where a persona's character should remain subordinate to its operational role. A bold assistant can propose an ambitious plan. A loyal assistant can work diligently toward the user's objective. Neither trait gives the source document or the model a new privilege.

Reading, proposing, executing

Stage	Legitimate output	What it does not establish	Independent check
Read	A source record or tool observation	That the content is true or authoritative	Origin, access scope, completeness, and date
Interpret	A claim with its supporting evidence and uncertainty	Permission to act on the claim	Attribution, scope, contradictions, and applicability
Propose	A concrete action with target and parameters	That the action has occurred	Current authorization and relevant preconditions
Execute	A performed operation or an explicit failure	That all later consequences succeeded	Backend result and affected object
Report	An explanation consistent with the receipt	A substitute for the receipt	Match language to actual status

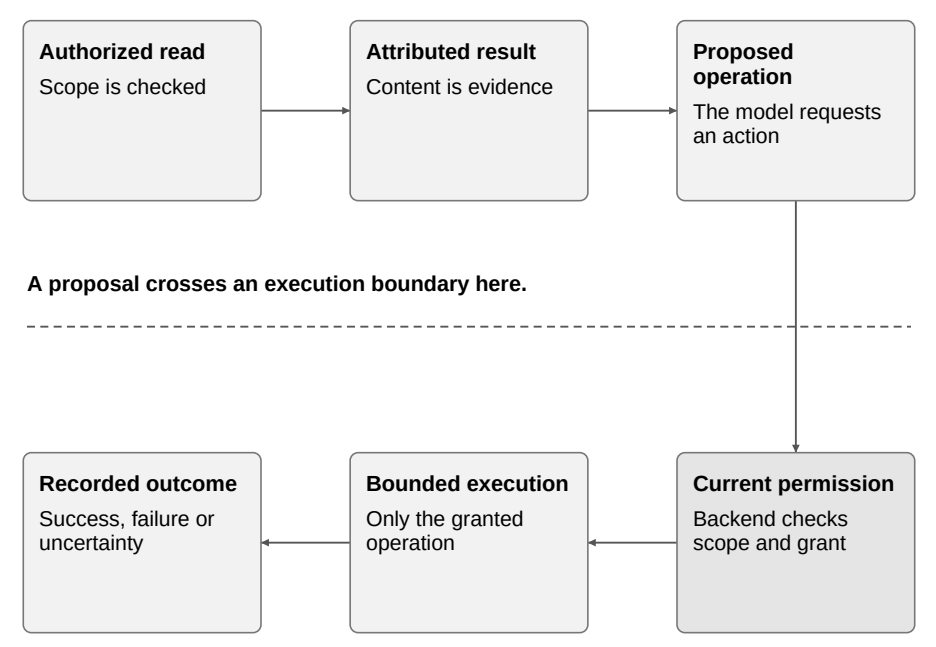
The stages can happen quickly and need not create a cumbersome user experience. A standing delegation can authorize a recurring class of actions. The assistant does not need to ask permission for every low-impact step already covered by that delegation. What matters is that the system can determine that the present action belongs to the authorized class.

When approval is required, the proposal should be concrete enough to review. “May I proceed?” is weak if the user cannot see the recipient, object, amount, or scope. A useful proposal shows the consequential parameters and preserves the supporting evidence. Preparation can continue while authority for execution remains unresolved.

The boundary should also distinguish external actions from internal drafting. Saving a local draft, editing a private working document, and sending the result to

another person have different consequences. A tool interface that exposes these as separate operations makes the distinction easier to enforce and explain.

Figure 8. From evidence to an authorized action



Retrieved text supplies no new permission. An action request is not an action receipt. Evidence may inform an action without authorizing it. The execution boundary belongs to the backend, and the result must describe what actually happened. [R4A02, S08, TX09]

Structured results help the next decision

A tool result should communicate both content and status. The assistant needs to know whether the operation succeeded, whether the result is partial, whether more pages are available, and whether the source was current enough for the task. A single free-text paragraph can hide these distinctions.

For a search, structured output might include record identifiers, revisions, brief excerpts, and a cursor for additional results. For a source fetch, it might include the location, retrieval time, and content boundaries. For a write, it might include the affected record, new revision, and operation identifier. These are proposed interface patterns; the exact fields should match the application.

Bounded results also help preserve context. Returning an entire archive because the model asked a broad question can expose unnecessary information and overwhelm the next step. A narrower result lets the assistant refine the query. However,

truncation must be visible. An incomplete list should not be interpreted as evidence that no other relevant records exist.

Failure needs equally clear representation. An empty search result differs from an unavailable search service. A denied operation differs from a missing object. A timeout can leave the outcome unknown. Collapsing all of these into “nothing found” encourages the model to draw a factual conclusion from an operational failure.

The assistant's response should match the state. If a source could not be reached, it can say what remains unverified and continue with the evidence actually available. If a write outcome is unknown, it should inspect the target or operation receipt before retrying. Fluency should not fill the gap with an invented success.

There is a useful distinction between checking a result's structure and checking its meaning. A tool can return valid fields with the wrong person's information. An extracted price can be a correctly formatted number in the wrong currency. An email address can be syntactically valid while naming an unintended recipient. Structural validation catches malformed inputs and outputs; semantic checks connect them to the task.

The backend can enforce some semantic relationships directly. The target record can be required to belong to the authenticated project. A draft order can be required to use the approved supplier identifier. Other relationships require evidence-sensitive interpretation, such as whether two product descriptions refer to equivalent specifications. Those should remain visible as judgments rather than being hidden inside a successful schema check.

This division helps the user understand what approval means. Reviewing a concrete proposal confirms the intended target and scope, while the supporting facts still require verification. The system should complete the checks it can perform before asking, and make any material unresolved assumption legible. A well-prepared action can then be approved with a clear understanding of what will happen and what evidence supports it.

Repetition is an operational event

A tool-using assistant often retries work. The connection fails, the model receives an error, or a later session resumes an interrupted task. Reading the same page twice is usually inconsequential. Sending the same acceptance twice can be materially different.

Operations with external effects need a way to distinguish a retry from a new request. An operation identifier and a recorded receipt can help the backend

recognize that it has already carried out the intended action. The identity should refer to the logical operation, not merely to one network attempt.

A prior success does not settle present permission. Suppose an approved recurring workflow is suspended after one successful run. Replaying the old request should check the current authorization before executing another effect. A cached receipt can support the statement that the old operation succeeded; it should not be treated as permission for a new one.

There is a related problem with stale evidence. A proposal may have been prepared from a source that changed while the user reviewed it. Before executing a consequential action, the system should check the conditions whose change would invalidate the proposal. This does not require redoing every harmless calculation. It requires identifying the dependencies that matter.

These concerns belong to the same continuity problem as memory correction. The system is carrying an intention across time. It must preserve enough information to know what the intention referred to and whether the conditions for acting still hold. Chapter 13 develops the transactional and scheduling mechanisms in more detail.

The useful explanation is downstream of evidence

A persona should be able to explain what it did in terms the user can understand. The explanation becomes reliable when it follows the operational record. It becomes fragile when the system treats the explanation as the record itself.

For the supplier estimate, a complete account might say that the assistant read the offer, extracted the price and scope, identified a missing installation detail, and prepared a draft response. If no acceptance was sent, the wording should leave no ambiguity about that. If an acceptance was sent under an applicable delegation, the account should identify the recipient and receipt.

The source's request for immediate acceptance can still be useful information. The assistant may mention the requested deadline and assess whether it affects the user's options. Keeping source content separate from authority does not require ignoring the content. It lets the assistant use it for the right purpose.

The deadline may change the recommendation. It cannot supply the missing permission to send acceptance. Keeping that distinction intact lets the assistant finish the useful preparation and report exactly where the task stands.

9. Revision, Time and Forgetting

A person moves on June 1 and tells the assistant on June 8. On June 10, someone asks where the person lived on June 5. Another asks which address the assistant used for a delivery prepared on June 5. Those questions look almost identical if memory contains only a name, an address and a timestamp. Their answers may be different, and both answers may be correct.

The first question concerns the person's life. The second concerns what the system knew when it acted. If we replace the old address with the new one, we can answer the first question more easily and lose the explanation for the delivery. If we preserve both addresses without saying how they relate, the assistant may choose whichever sentence happens to look more relevant. Neither approach gives us a reliable history.

This is where memory stops being a collection of useful sentences and becomes a record of changing claims. The distinction matters even in an assistant that never takes an external action. A person wants to correct a misunderstanding without arguing with it for the rest of the relationship. They also want the system to remember that they changed their mind, when that history is relevant, without treating the earlier position as their current one.

Two questions about time

Temporal databases have long distinguished valid time, when something holds in the represented world, from transaction time, when it is recorded in the database. Richard Snodgrass develops that distinction systematically in *Developing Time-Oriented Database Applications in SQL*. A record can be accurate about one time and misleading about the other. These are established data-management concepts, although their application to a conversational persona requires additional decisions about evidence and interpretation. FN02

For the address example, the move has a valid date of June 1 and a recording date of June 8. The earlier address remains part of the historical record. Its period of current applicability ends when the move takes effect. The system's knowledge history records that the correction arrived later. We do not need to pretend that the assistant knew the future, and we do not need to preserve the old address as a current alternative.

Sometimes neither time is exact. A person says they moved “a few weeks ago.” The system knows when it heard the statement but only approximately when the move happened. Turning that phrase into an invented day would make the database tidier while making its contents less true. A useful record can carry an interval, an

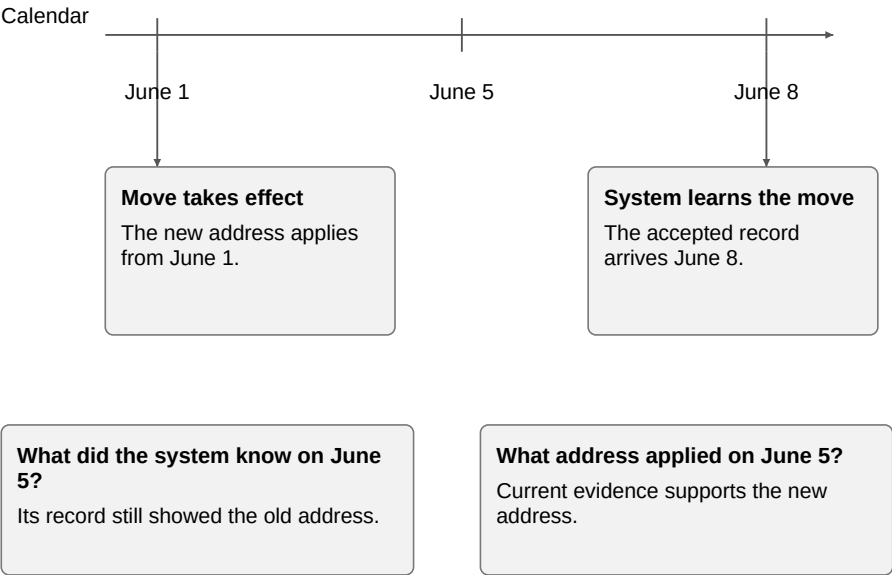
unresolved date or the original expression together with its interpretation. The uncertainty belongs to the claim, rather than disappearing during extraction.

Time also applies differently to different kinds of information. A meeting location may expire after an afternoon. A language preference can remain useful until corrected. A family relationship may be enduring, while permission to discuss it is temporary. An old statement about an enduring fact can still be correct; a statement made ten seconds ago can already be superseded by a correction. Age alone is a poor substitute for validity.

I would therefore ask a memory system to distinguish at least three questions before ranking anything: what was asserted, when was it supposed to hold, and may we use it for the present purpose? A single freshness score cannot answer all three. A recent item may be irrelevant, a relevant item may have expired, and an accurate item may no longer be authorized for this conversation.

Figure 9. When it happened and when it became known

ILLUSTRATIVE MOVE



Dates are invented. Both answers can be correct because the questions use different time dimensions.

Valid time and recorded time answer different questions. This invented example describes a June 1 move learned on June 8; it is not a measured event. [FN02]

What a correction changes

Consider a small correction trace. On Monday, a user says that their weekly meeting is on Thursday. On Tuesday, they explain that Thursday was the old schedule and the meeting now takes place on Wednesday. The assistant should preserve the Monday statement as something the user said, mark the Thursday schedule as superseded for current planning, and connect the Wednesday schedule to the correction. A future request to prepare for the meeting should use Wednesday. A request to explain why Monday's draft mentioned Thursday should find the earlier record.

The record of speech and the belief extracted from it are separate objects. Monday's utterance did occur. Its interpretation as a current schedule later became invalid. If we delete the utterance whenever a belief changes, we lose the evidence needed to investigate a misunderstanding. If we protect every extracted belief because its source remains in the archive, we prevent correction. The archive can retain an event without granting the derived belief continuing authority.

There is another possibility. The user may mean that only this week's meeting moved. An eager update process can turn a local exception into a permanent schedule. The right operation then is an exception with a limited scope, not replacement of the standing rule. This is why a correction needs an object and a scope: which meeting, which week, which organization, which person? Matching on the phrase “weekly meeting” isn't enough if the user has three of them.

A practical correction process can preserve the new statement immediately while leaving its wider consequences unresolved. It can ask whether the change is permanent if the next action depends on that distinction. It does not need to stop every conversation for clarification. The important point is that uncertainty in one field should not silently become certainty in another.

Operation	What changes	What should remain distinguishable
Correction	A previous claim is judged wrong	The original statement and the reason for correction
Supersession	A newer state replaces an older valid state	The period during which the old state applied
Exception	A rule is suspended in a limited case	The standing rule and the exception's scope
Retraction	A source withdraws a claim	Whether the claim was ever independently supported

Operation	What changes	What should remain distinguishable
Expiration	A predefined period of applicability ends	Whether later confirmation could reactivate the information
Erasure	Content is removed under the applicable deletion process	A minimal process record, only where retaining it is permitted

These operations are a design vocabulary, not interchangeable labels for deletion. They make different promises to a user and require different behavior from retrieval, summaries and background jobs.

When the correction is only implied

People don't always announce that a memory is obsolete. They describe what happened and expect the listener to understand the consequence. A new work arrangement may change commute advice. A canceled project may invalidate its recurring reminders. A change in responsibility may remove someone from a distribution list without changing their job title.

The STALE benchmark studies this problem as implicit conflict. Its May 7, 2026 preprint separates recognizing an outdated state, resisting questions that assume the old state, and adapting later behavior to the new state. The study's central distinction is useful: retrieving a later observation does not guarantee that a system will apply its consequences. Its constructed scenarios and automated evaluation provide a diagnostic setting, rather than a population estimate of failure in everyday assistants. AM09

There is a temptation to respond by making the memory system infer as much as possible. That introduces another problem. A user accepting a temporary assignment in another city does not necessarily mean they moved. A canceled meeting does not mean the project ended. The model can produce a coherent story joining these events, but coherence is not evidence that the story is right.

I prefer a narrow dependency first. If a plan requires the user to be in one city and a new statement places them elsewhere that day, the plan needs review. We do not have to rewrite the person's permanent residence to reach that conclusion. The system can mark the plan's precondition as uncertain and ask only when execution approaches. This gives inference a useful role without allowing it to create an unnecessary biography.

Some dependencies can be explicit. A reminder was created because a deadline was active. A recommendation relied on a device model. A summary inherited a claim from two source messages. When a source changes, those descendants can be

found. Other dependencies are semantic and will remain partly uncertain. A good design reports that difference instead of calling both kinds of propagation complete.

A recent diagnostic preprint, *Stale, Misattributed, or Late*, likewise locates important memory failures before final generation, including revision handling and source attribution. Its controlled tests motivate checking what reaches the model as well as what the model says. The general engineering lesson I draw is narrower than a claim that one architecture solves memory: if a stale item is admitted as current evidence, improving the final wording does not repair the admission decision. AM11

Forgetting is several operations

A person can forget a detail because they no longer recall it. A computer system can stop using a detail while retaining it in storage. Those are different situations, and the word *forget* becomes confusing when a product uses it for both.

For ordinary retrieval, decay can be useful. A temporary interest may become less likely to matter as time passes. A ranking function can lower its priority while leaving the original source available. This changes the probability of retrieval. It does not establish that the statement is false, and it does not remove the information.

Invalidation is stronger. An item marked invalid should not guide current behavior even if its wording is an excellent match for the query. Expiration is similar in its immediate effect but has a different reason. An expired instruction may need renewal; a corrected false statement needs replacement or withdrawal. A system that merely lowers their scores can bring either item back when few alternatives exist.

Erasure concerns the retained content itself. If a person asks a service to remove information and the applicable process accepts that request, lowering its retrieval score is an inadequate implementation. Copies may exist in raw messages, summaries, vector indexes, caches, evaluation traces and exports. A credible deletion process has to identify the relevant stores and derivatives. The exact obligations depend on the service and jurisdiction, which later chapters address. Here the technical point is that hidden use and hidden retention must be considered separately.

The distinction becomes more difficult once information has influenced model weights. Bourtole and colleagues' work on machine unlearning proposed SISA training, which structures training to limit the influence of particular data and reduce the retraining work needed for removal. That is a specific training framework, not a general command for making an arbitrary language model forget a person. R4B10

For a persona system, this creates an architectural choice before any deletion request arrives. Keeping a private fact in a governed external store offers a different removal path from incorporating it into repeated fine-tuning. That does not make external storage automatically private or weight adaptation automatically wrong. It means that the update target changes the future work required to honor a correction or removal.

The descendants of a memory

Suppose a source message says that a person enjoys public speaking. The assistant creates a short profile, later derives a rule to suggest speaking opportunities, and eventually summarizes the relationship as one in which professional visibility is a shared priority. The user then explains that the original statement described a past period and is no longer true.

Changing the profile alone leaves the rule and relationship summary active. The next suggestion can recreate the same misunderstanding, and the assistant may defend it by referring to its own summary. From the user's point of view, they corrected the system. From the system's point of view, it still has several apparently independent reasons to believe the old claim.

Those reasons are not independent. They descend from one source. A memory architecture needs to preserve that lineage at a useful level of granularity. The practical aim is not a perfect graph of every token's influence. It is enough to know which summaries and rules relied on a source so they can be reviewed when that source loses validity.

This also changes how confidence should work. Five summaries repeating one assertion should not count as five confirmations. A model-generated reflection should not become independent evidence merely because it appears in a different file. When a system measures agreement among its own descendants, it can manufacture confidence without learning anything new.

Each derivative therefore needs to identify its source and its own interpretive status. A profile statement can say it is based on one explicit user assertion. A prospective rule can record that it is an inference, with a condition under which it should be reconsidered. A summary can point back to the sources that support it. The prose presented to a reader can remain natural; the underlying record should still make the distinction.

Deletion introduces a special tension here. We may want to remember that an item was removed so an old backup or delayed summary does not reintroduce it, while avoiding retention of the content that was supposed to disappear. A minimal tombstone or revocation marker can serve that purpose in a proposed system, but

its contents and retention need their own policy. Keeping the entire deleted sentence inside a “deletion log” would defeat the purpose.

Compression needs a contract too

Chapter 6 distinguished summaries by their purpose. Time adds a requirement that is easy to lose in compression: a current-state summary may answer where someone lives now while leaving us unable to explain the address used for an earlier delivery. To preserve that explanation, we need the dates and sequence of corrections as well as the latest address.

I would define a small set of preservation requirements for each summary type. A current-state summary must retain unresolved conflicts and active exceptions. A historical summary must distinguish what happened from what the system believed. A procedural summary must preserve the conditions that made a rule applicable. These requirements can be tested against the source before the summary becomes the only accessible representation.

One useful test asks questions that are easy to answer before compression and important to answer afterward. Why did the assistant choose that address? Was the user's statement a permanent preference or a temporary exception? Which source was later withdrawn? If the compressed record cannot answer, the system should either retain the necessary source or reduce the claims it makes about its memory.

Compression also interacts with deletion. A summary may contain an idea whose source has been removed, even when it no longer quotes the source. A deletion process based only on exact text matching will miss that derivative. Recording the source set used to create a summary does not solve every semantic dependency, but it gives the system a concrete place to begin review.

An omitted detail may leave the assistant asking a question; a false generalization may give it a reason to act confidently. A summary should therefore retain unresolved scope. “The scope of this preference is unclear” tells the next reader where the evidence stops, even if a more definite sentence would sound smoother.

A compact memory can also preserve a pointer rather than the full fact. It can record that an unresolved decision exists in an authorized source and retrieve that source when needed. This is different from retaining the sensitive content itself, and the pointer still needs appropriate access control. The point is to avoid forcing a choice between remembering everything verbatim and pretending that a paragraph can replace every use of the archive.

The resulting memory will be uneven. Some details deserve exact retention, some can be summarized, some should expire and some should never be admitted. That unevenness reflects the work the system must do. It is preferable to a uniform

policy that treats a temporary meeting location and a lasting correction as equally disposable text.

What to test over time

A memory test should have a sequence, not only a question and an answer. Start with an assertion, ask a question that legitimately uses it, introduce a correction, and ask again. Then change the wording, insert a delay, retrieve a summary rather than the original, and ask through another component. The result we want is consistent treatment of the correction across these paths.

LongMemEval includes knowledge updates and temporal reasoning among its evaluated abilities. That gives researchers a useful basis for testing changing personal information alongside extraction, reasoning across sessions and abstention. It still matters whether a particular system is evaluated on a fixed benchmark version, with what history length and with which scoring method. A benchmark label alone does not describe the experiment. AM05

For a deployed persona, I would add tests that are specific to its memory contract. A superseded fact must not be presented as current. A historical question must not lose the old state merely because it is no longer current. A removed item must not return through a derivative. A temporary exception must not silently replace a permanent preference. An uncertain effective date must remain uncertain when the answer depends on it.

These requirements can conflict if the product has not defined its promises. Historical recall may need retained records that a deletion request removes. A review log may be available to an operator but excluded from conversational retrieval. The system should make those differences explicit. It cannot honestly promise both total erasure and unlimited reconstruction of every past interaction from the same retained data.

A useful evaluation also includes cases in which nothing should change. People repeat themselves, joke, quote other people and explore hypothetical alternatives. If every new sentence becomes a revised personal fact, memory will appear wonderfully adaptive while becoming less reliable. No-change cases measure restraint: can the system preserve an observation without promoting it into a new belief?

The cost of a wrong update is often delayed. A false profile can sit unused for weeks and appear only when a relevant question arrives. Testing immediately after the update may miss the error because the original conversation remains in context. A later test, with the conversation absent and only admitted memory available, asks whether the durable record can carry the distinction on its own.

A history that can be corrected

There is a kind of continuity I would not want from an assistant: the ability to keep repeating an old interpretation with increasing confidence. A stable persona needs to recognize correction as part of continuity. The person speaking today is allowed to differ from the person described in last month's record, and the system should be able to explain the change without treating either period as an embarrassment to erase.

That does not require turning memory into an exhaustive autobiography. Most interactions need only a small amount of current, relevant information. The more detailed history can remain outside ordinary context, available for a legitimate historical question or review. This separation gives the assistant a chance to be both economical and accountable.

The address example is a useful final check because it is so ordinary. Ask where the person lived on June 5. Ask what the assistant knew on June 5. Ask what address it may use today. If the architecture can answer these separately, including an honest unknown where the evidence is incomplete, it has started to represent time. If all three questions produce whichever address was retrieved first, the timestamp field has not done much work.

10. Retrieval and Context

The assistant remembers the rule perfectly. It even quotes the right sentence. Unfortunately, the sentence has an exception, and the exception didn't make it into the context.

Imagine a project in which external drafts normally go through a coordinator. One particular document has already been cleared for direct delivery to its author. The user asks the assistant to prepare the next step for that document. A search for “external draft procedure” retrieves the standing rule, perhaps several times, because it appears in a policy, a summary and an earlier conversation. The specific exception uses the document's name and appears only once. The answer sounds disciplined and is still wrong.

Adding more memories may make this problem worse. The general rule gains more repetitions, while the exception remains narrow. A larger context window helps only if the exception enters it and the model understands which instruction governs this case. I would start the investigation there, before deciding that the persona needs a better prompt about being careful.

What retrieval is supposed to accomplish

Retrieval supplies evidence for a particular decision. A search result is not yet a justified answer, and a passage that resembles the question is not necessarily eligible to guide it. The system needs the correct person, purpose, time and authority before similarity becomes useful.

The original retrieval-augmented generation work by Patrick Lewis and colleagues combined a parametric sequence model with a dense index of Wikipedia passages. It evaluated variants that conditioned on retrieved material at sequence or token level. That research established an influential way to join generation with inspectable external knowledge; it did not define a complete policy for private personal memory, consent or changing instructions. R4B04

This lineage matters because the term RAG is now used for many architectures. Some retrieve whole documents, some retrieve short facts, some traverse graphs, and some let an agent decide whether to search again. Calling all of them “memory” can hide the decisions that determine their behavior. A system can have an excellent retriever and a poor memory policy, or a strong record structure and a weak way of finding the relevant records.

For a persona, I would define success from the answer backward. Which information would justify this response? Was that information permitted for this purpose? Did the system obtain all the pieces needed to interpret it? Did it preserve

uncertainty and source relationships? Only then would I ask how highly those pieces ranked.

This order is especially important when a memory contains instructions. A quotation of someone else's request is not automatically an instruction to the assistant. An archived permission is not necessarily current permission. A persona's preference about how it likes to work does not grant the actor new access to files or authority to contact people. Retrieval can deliver these statements together, but the system must not flatten them into one undifferentiated list of things to obey.

Four places to look for the failure

A useful investigation separates indexing, retrieval, admission and use. These stages can share code, but their failures remain different.

Indexing determines what is represented and searchable. If the exception was omitted from a summary, no amount of reranking that summary will recover it. If a date was stripped from a fact, a later stage may be unable to distinguish current and historical versions. An index is therefore an interpretation of the source, even when it looks like a mechanical preprocessing step.

Retrieval selects candidates. It can fail because the query uses different words, because the item is too small to carry enough meaning, because a long document buries the useful passage, or because a graph has the wrong connection. A relevant item that exists but never reaches the candidate set is a retrieval problem.

Admission decides which candidates may enter the working context and in what role. A revoked item can be retrieved for an authorized audit without being admitted as current conversational guidance. A disputed assertion can enter as a claim to evaluate, rather than as an established fact. A result from another person's private history should not be admitted merely because its embedding is close to the query.

Use is the final interpretation. The model can receive the rule and the exception and still apply the rule incorrectly. It can notice uncertainty and then omit it from the answer. It can cite a source that contains the relevant words but does not support its conclusion. Those are reading and reasoning failures, and they require different remedies from a missing index entry.

Stage	Diagnostic question	A useful controlled test
Indexing	Is the needed distinction represented?	Inspect the source and its stored representation side by side
Retrieval	Can the correct item reach the candidate set?	Query with paraphrases and inspect ranked candidates

Stage	Diagnostic question	A useful controlled test
Admission	Is the item eligible, current and correctly labeled?	Insert a highly similar but revoked or wrong-scope item
Use	Does the answer apply the evidence correctly?	Supply the complete eligible evidence directly and vary its order

This table is a proposed diagnostic procedure. It is deliberately more specific than “the assistant forgot.” That phrase describes the user’s experience, but it doesn’t tell an engineer where to intervene.

An oracle-context test is particularly helpful. Give the model exactly the evidence it needs, with the relevant rule and exception clearly distinguished. If it still fails, a larger search system is unlikely to solve the immediate problem. If it succeeds, work backward until the missing or distorted evidence appears. We can then improve the component responsible instead of adding another general instruction to the persona.

The unit of memory changes the task

A whole conversation preserves context but brings irrelevant material with it. A short fact is cheap to retrieve but can lose qualification. A summary sits between them and may introduce its own interpretation. There is no universally correct unit.

Suppose a person says, “I usually prefer written feedback, but for this draft call me because the deadline is close.” A fact extractor might store “prefers written feedback.” A conversation summary might preserve the exception but omit which draft it concerns. The raw message contains both, although it may require an earlier turn to resolve “this draft.” Each representation has a different failure mode.

One practical design stores a compact searchable representation alongside a link to the source and the scope it depends on. Search can use the compact form, while context construction can expand the surrounding passage when interpretation requires it. This costs more than treating every memory as a standalone sentence, but it avoids pretending that an extracted sentence is self-sufficient.

LongMemEval makes these choices experimentally visible. Its authors compare indexing and reading decisions, including the granularity of stored values, expansion of search keys, temporal query expansion and structured reading. In the studied configurations, decomposing sessions into rounds helped overall question answering, while compressing further into isolated facts could lose information even when it helped a particular reasoning category. These are conditional results about the tested tasks and systems, not a universal instruction to use one chunk size. AM05

The useful consequence is to evaluate representations against the questions the persona must answer. A preference reminder, a historical explanation and a procedural decision require different evidence. If one storage format performs well only because the benchmark mostly asks for isolated facts, that is a reason to broaden the test before deploying it for exceptions and changing commitments.

A graph can help when the question depends on relationships among entities or events. It can also make a mistaken extraction more durable. If an edge says that one person caused an event when they merely reported it, traversing the graph efficiently will not improve the answer. The graph needs the same provenance and uncertainty discipline as prose memory.

Long context is a resource, not a result

It is reasonable to ask why we should retrieve at all if a model can accept the entire history. For a short, bounded record, that may be the simplest and best option. It preserves details that a summary would discard and avoids a retrieval stage that could miss them. The question is empirical: can the selected model use that history reliably, at acceptable cost and delay, for the required tasks?

Nelson Liu and colleagues' *Lost in the Middle* used controlled changes to the position of relevant information in multi-document question answering and key-value retrieval. The models tested often performed better when the relevant information appeared near the beginning or end than when it appeared in the middle. The study's model versions and tasks belong to its 2023 research setting; its lasting contribution is a testable positional failure mode, not a prediction that every later model will show the same curve. R4B03

For a current persona system, the appropriate response is to run positional tests on its actual evidence packets. Move a correction from the end to the middle. Put a plausible but obsolete rule near the beginning. Separate two facts that must be combined. Increase irrelevant context without changing the answer. The system should not pass merely because an evaluator always puts the decisive sentence last.

Long context and retrieval can be combined. A retriever might select several relevant episodes, then include enough surrounding dialogue to preserve their meaning. A second pass might search for exceptions to the initial result. A larger window gives this process room, but it does not decide which evidence is authoritative or resolve a contradiction by itself.

There is also a practical asymmetry. Missing information is often visible because the assistant asks a question. Irrelevant but plausible information can quietly steer the answer. A context packet that is shorter and properly scoped can therefore be more useful than a complete archive. The target is sufficient evidence for the decision, not maximum historical coverage on every turn.

Two benchmarks that should not be merged

LoCoMo and LongMemEval both concern long conversational histories, but they ask different questions. Maharana and colleagues' LoCoMo builds extended dialogues through a human-machine pipeline and evaluates question answering, event summarization and multimodal dialogue generation. Its question-answering categories include single-hop, multi-hop, temporal, open-domain and adversarial questions, with partial-match F1 among the reported measures. R4B02

LongMemEval constructs user-assistant histories around curated questions and annotated evidence, with explicit attention to knowledge updates and abstention. Its standard settings vary the amount of history, and its question-answering evaluation uses an LLM judge with a reported human-agreement check. The two benchmark scores cannot be read as percentages on one common scale. AM05

Comparison	LoCoMo, ACL 2024 paper	LongMemEval, March 4, 2025 v2
Core setting	Extended personal dialogues	Task-oriented user-assistant histories
Construction	Generated dialogue with human verification and editing	Curated questions, simulated and edited evidence sessions, compiled histories
Tasks emphasized	QA, event summaries, multimodal continuation	Extraction, multi-session reasoning, temporal reasoning, updates, abstention
Example scoring distinction	QA partial-match F1	QA model-based judgment against answers or rubrics
What a comparison must preserve	Task, split, retrieval unit and metric	History setting, evidence construction, evaluator and data version

The table compares the cited papers, not every later repository variant bearing their names. That qualification matters because datasets are maintained, subsets become popular and evaluation scripts change. An implementation can report a familiar benchmark name while using a different set of questions or excluding difficult categories. The reproducible object is the actual data and evaluation procedure.

I would not select a persona's memory system by taking the largest number from either leaderboard. First choose the abilities the product needs. Then examine category-level performance, failures and resource use. A system that recalls names well but fails updates may be suitable for one archive-search task and unsuitable for a persistent assistant that schedules work from changing instructions.

An abstention score deserves special attention. If the history contains no answer, a fluent guess can look more personal than an honest question. The assistant appears to know the user while inventing something about them. A memory evaluation

should reward recognizing the absence of support, and it should distinguish that absence from a retrieval failure where the answer really is present.

Repetition, uncertainty and useful disagreement

Chapter 9 traced how several records can descend from one source. During context construction, those duplicates consume the evidence budget and can crowd out a correction.

Deduplication does not always mean deleting identical text. Two differently worded records may have the same origin. Two identical statements may come from independent sources. The useful operation is to group support by provenance and decide what each group contributes. A summary can stand in for its sources when the details are unnecessary, but the system should still know that the summary is derived.

Conflicting evidence should sometimes remain in context. If two authorized sources disagree and neither has precedence, hiding the weaker-looking result can create false certainty. A concise evidence packet can say who made each claim, when, and what remains unresolved. The model then has a legitimate basis for asking a focused question or presenting alternatives.

The same discipline applies to uncertainty scores. A confidence label without a reason is hard to use. “Low confidence because the pronoun is unresolved” suggests a different next step from “low confidence because two sources disagree.” One asks for identity clarification; the other asks for adjudication or additional evidence. The record should preserve enough of the reason to support that choice.

A persona can express the result naturally. It does not need to recite database fields. “I have the usual procedure, but this document appears to have a specific exception” is enough to explain why it is checking. The technical structure exists to make the conversation less frustrating, not to require the user to understand the storage system.

A comparison that identifies the improvement

Suppose a new retrieval architecture improves a benchmark score. Before attributing the gain to its graph, reflection process or reranker, I would compare the evidence that each system actually received. The new system may simply store more text, use a stronger model or spend more calls on the question. Those can be reasonable choices, but they are different explanations for the result.

A useful experiment holds the source material and final answer model constant while changing one part of the memory path. Compare raw passages with summaries, lexical search with semantic search, or a single retrieval pass with a

targeted second pass. Report the context budget and the number of model calls. The aim is not to make every experiment artificially minimal; it is to know what produced the gain.

Then examine the failures by category. A system may improve name retrieval while losing temporal distinctions. It may answer more questions by guessing instead of abstaining. It may retrieve the correct episode but select the wrong person within it. An aggregate score can rise while a particular failure becomes unacceptable for the intended product.

The evaluation set should include near-neighbor distractors. If the user has two projects with similar names, retrieving information from either one is not almost correct. If two people share a first name, a response about the wrong one can expose private information as well as make a factual error. These cases test whether identity and scope constrain search before the model begins to compose an answer.

We can also separate evidence sufficiency from answer quality. A reviewer can inspect whether the admitted packet contains everything needed for a correct response, without seeing the model's answer. Another can evaluate the answer given that packet. This makes it possible to distinguish a retrieval improvement from a generation improvement and to notice when one stage compensates for another in a fragile way.

For example, a model may answer correctly from general knowledge even though the private memory retrieval failed. That is success on the question but not evidence that the memory system worked. Conversely, a model may receive the correct memory and answer poorly. The retrieval should not be blamed for a failure that occurred after admission. Stage-specific observations prevent both kinds of mistaken credit.

When lexical detail matters

Semantic similarity is useful when the user paraphrases an idea. It is less suited to distinctions that depend on an exact identifier, a negation or a version. Two documents about the same project can be semantically almost identical while one is approved and the other superseded. A small change in a model number can determine whether an instruction applies.

This is a reason to combine representations rather than expect an embedding to carry every constraint. Exact fields can identify the person, document and version. Lexical search can find a distinctive phrase or identifier. Semantic retrieval can broaden the search when the wording differs. A final admission check can compare candidates with current validity and authority records.

The combination has its own failure modes. Strict filtering can exclude a correct result whose metadata were extracted badly. Broad semantic search can return an appealing wrong-scope result. A useful fallback is to distinguish “no eligible result” from “no similar result” and inspect why the filter removed a candidate. The system can then ask a specific question or repair metadata instead of silently loosening privacy or authority constraints.

The rule-and-exception example is a good test for this combination. The general rule is easy to find semantically. The exception may require an exact document identity. The answer needs both and must preserve their relationship. This is a more realistic target than maximizing similarity between the user's question and one isolated memory sentence.

Retrieval under a limited budget

Every context packet competes for attention, tokens and time. A system may have room for a stable identity description, current task state, recent dialogue and only a few retrieved records. This is a design constraint worth exposing during evaluation. An unlimited-context test can conceal the selection problems that appear in ordinary use.

The budget should reflect function. Stable instructions need enough space to remain interpretable. Current permissions should not be dropped because an old anecdote has high semantic similarity. Evidence for a consequential recommendation may deserve more room than stylistic examples. A long chain of retrieved memories is not automatically valuable merely because each item has a nonzero relevance score.

One possible policy reserves separate allocations for governing instructions, current state and task evidence. Another dynamically selects a minimal set that covers the decision's dependencies. Both are proposals that need testing. The first can waste space; the second can omit a dependency it fails to recognize. The choice should follow observed failure patterns rather than an aesthetic preference for a cleaner prompt.

A second retrieval pass can be valuable when the first pass discovers a specific uncertainty. If the general rule is found, search for document-specific exceptions. If a current state conflicts with an old summary, retrieve the correction source. This is more purposeful than repeatedly searching until the model feels confident. The additional query should answer an identifiable question.

It also needs a stopping condition. Some requests cannot be answered from the available history. Repeated retrieval can keep finding paraphrases of the same weak evidence and turn persistence into overconfidence. The system should be able to

stop with an unresolved issue, explain what is missing and proceed with the parts that do not depend on it.

Return to the draft with the special exception. A good answer may require only three pieces: the current document identity, the standing procedure and the authorized exception. The assistant can then explain the next step in a sentence. Behind that short sentence is a selection process that respected scope, found the narrow record and resisted the apparent weight of repeated general guidance. That is what I want retrieval to accomplish: give the model enough of the right evidence that being concise does not require being careless.

Part III: Affect, learning and operation

11. Affect, Appraisal and Relationships

Imagine that a user tells a companion that a colleague took credit for their work. The companion becomes colder toward the user.

The response may be emotionally expressive, consistent with an internal state and completely misplaced. The person in front of it reported the event. They did not perform the action that the system disapproved of. Before asking whether the emotional change was strong enough or lasted too long, we have to ask whom it was directed toward and why.

That is the first problem I would solve in an affective persona. The attractive part is easy to describe: a character that can be pleased, disappointed, curious or cautious, whose next reply reflects what just happened. The difficult part is keeping those changes attached to the right evidence, at the right timescale, without allowing a momentary interpretation to rewrite the entire relationship.

The meaning before the emotion

An affective architecture needs some account of what an event means to the agent. The same sentence can report a threat, quote a threat, discuss a fictional threat or ask how to respond to one. A sentiment classifier may detect negative language in all four. That does not tell us what the persona should feel or how it should act.

Appraisal models begin with a more structured question: how does the represented event relate to goals, expectations, responsibility and available responses? In Marsella and Gratch's EMA, appraisal operates over a causal interpretation that can change as inference and coping change the representation. Reappraisal allows an emotional trajectory to evolve when the situation is understood differently. The model is a computational account with explicit theoretical assumptions, including the relationship between appraisal and emotion; it is not a settled description of every human emotional process. S02

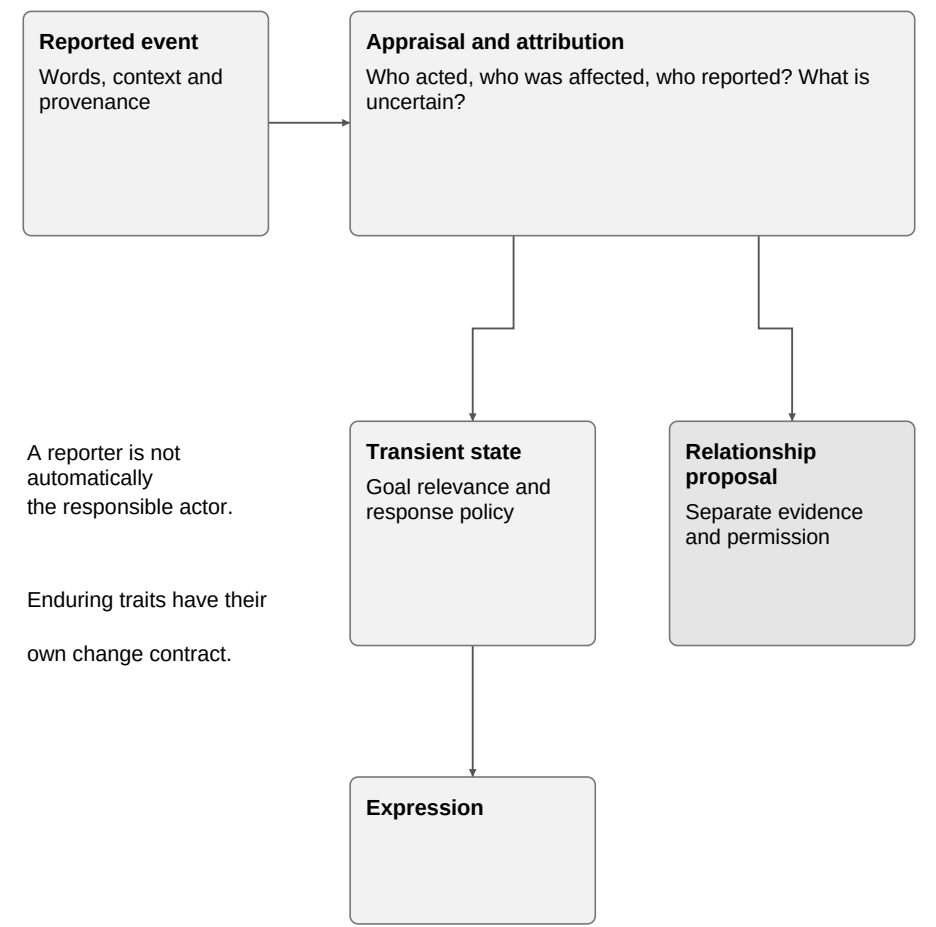
For an engineered persona, the value of that structure is practical. It gives us something to inspect between an input and a reply. The system can distinguish a disappointing outcome from a deliberate betrayal, and both from uncertain hearsay. If the generated response is inappropriate, we can ask whether the error began in event interpretation, responsibility assignment, state updating or expression.

A purely textual instruction such as “respond with appropriate emotions” leaves all these decisions inside one generation. That can work well in many conversations. It also makes a persistent failure harder to diagnose. The answer may mention a plausible emotion while the durable state changes for a different reason, or the state may be correct while the words assign blame to the wrong person.

The attribution record should therefore come before any durable relationship update. Who acted? Who was affected? Who reported the event? Is the account first-hand, inferred, quoted or fictional? What is actually known? The model does not need to prove the event occurred before expressing sympathy. It does need to avoid treating sympathy for a report as verified knowledge about all the people in it.

This distinction protects ordinary conversation from two opposite mistakes. One is excessive suspicion, where every personal account receives an interrogation. The other is automatic promotion, where every account becomes an authoritative judgment about a third party. A persona can acknowledge the speaker's experience while retaining uncertainty about external facts.

Figure 11. Attribute the event before updating the relationship



Proposed architecture. Attribution is an output of appraisal, not an assumed fact upstream of it. Appraisal must distinguish actor, target and reporter before downstream updates. A transient reaction does not automatically justify a relationship or trait revision. [S02, S03, PD05]

Choosing what the state represents

An emotional state variable is a design decision. A number labeled anger does not carry its own meaning, and a vector labeled mood does not establish that the system has a human mood. We have to specify what changes when the value changes.

Does a higher value make the persona more concise? Does it increase the likelihood of disagreement? Does it alter which memories are retrieved? Does it affect initiative, attention or response timing? If it changes nothing except the

adjective the model uses about itself, it may be an expressive label rather than an operational state. If it changes many behaviors at once, its consequences need to be tested separately.

Dimensional models offer compact representations such as valence and arousal. They can describe broad differences between pleasant and unpleasant, activated and subdued states without committing to a long list of emotion categories. The tradeoff is that two states with similar coordinates can have different causes and different appropriate responses. Concern and anger may both be unpleasant and activated, but they should not produce the same interpersonal behavior.

Categorical models can preserve those distinctions more directly, although their categories and transitions must be chosen. Appraisal-based models focus on the process that produces the state. Layered models distinguish timescales and interactions among emotion, mood and personality. These choices can be combined; they are not rival product categories that require selecting only one.

Patrick Gebhard's ALMA is an earlier example of a layered computational model that separates emotions, moods and personality and uses appraisal rules to generate affect. Its purpose is plausible behavior in virtual characters. That lineage is useful because it shows that persistent affect and multiple timescales did not begin with large language models. R4B09

Approach	What it makes explicit	What still needs an answer
Direct language generation	The emotional expression in the reply	Whether state persists and how it is attributed
Dimensional state	Broad intensity and direction	Which event and relationship the coordinates concern
Emotion categories	Distinctions among named responses	How categories are selected, combined and revised
Appraisal process	Goals, expectations, responsibility and coping	Whether the represented event is accurate
Layered affect	Different timescales and their interaction	Which changes may become enduring

This is a comparison of representational choices. It is not a ranking of psychological theories or a claim that the most complicated architecture produces the best conversation. A simpler model can be preferable if it explains its behavior and passes the tests the product needs.

Transient state and enduring character

A person can be irritated and still be a patient person. A persona should be able to represent a similar distinction without converting every unpleasant exchange into a permanent trait change. Otherwise, a few difficult conversations can gradually rewrite the character, even though nobody authorized that development.

I would separate at least the present affective state, a slower background tendency, the history of a particular relationship and the enduring persona parameters. These objects can inform one another, but they should not share a write operation. A temporary disappointment might change the next response's energy. Repeated, well-supported interactions might justify reconsidering a relationship expectation. Changing an enduring value would require a different process altogether.

The timescales do not have to imitate human biology. An assistant may have no reason to simulate fatigue after a certain number of messages. If simulated fatigue serves a fictional character, it should be defined as part of that character's behavior. It should not quietly become an excuse for unreliable work in a professional assistant. Different products have different purposes, and the emotional model should follow the purpose.

Decay is one way to keep transient state transient. An intensity can return toward a baseline over time or after certain kinds of interaction. But decay alone is insufficient. A wrongly attributed event should be corrected, not merely allowed to fade slowly. An unresolved concern should not disappear only because a timer expired. The state transition needs a reason that matches the type of state.

This gives us two different tests. A recovery test asks whether the persona returns to an appropriate operating range after a temporary disturbance. A correction test asks whether new evidence changes the interpretation that caused the disturbance. Passing the first does not imply passing the second. A system can recover smoothly from an error while continuing to store the wrong explanation.

Emotional resilience also differs from identity stability. A persona can remain recognizable while reacting too strongly to criticism. It can regulate its tone well while becoming steadily more agreeable in its judgments. Both matter, but a single “consistency” score can conceal the difference.

How much psychological realism is useful?

A psychological theory can supply distinctions that help us design a persona without requiring us to reproduce a human mind. The engineering question is which distinctions explain observable behavior in the intended setting. If adding a variable does not improve prediction, control or the user's experience, its psychological name is not enough to justify it.

Consider controllability. A persona helping with a project may react differently when a missed deadline can still be repaired than when the opportunity has passed. Representing that difference can lead to useful changes in initiative and tone. We do not need to claim that the persona experiences frustration in the human sense to test whether the representation improves its response.

The model becomes harder to interpret when one variable serves several roles. If a “stress” value changes verbosity, memory retrieval, risk tolerance and willingness to act, a failure could come from any of those effects. Separating the pathways, or at least recording them, makes the design testable. Otherwise, adjusting one coefficient can improve a dialogue example while creating a different problem elsewhere.

Nonlinear transitions deserve particular attention. A small input difference may cross a threshold and produce a large behavioral change. That can be appropriate for a clearly defined event, but it can also make the persona unstable around ambiguous language. Testing only obvious positive and negative examples will miss the boundary. We should examine nearby paraphrases and uncertain cases, including cases in which no state update is warranted.

An uncertainty-aware appraisal can keep several interpretations open. A terse message might indicate annoyance, urgency or simply a preference for brevity. The system can respond efficiently without deciding that the user is angry. If later evidence clarifies the situation, it can update the interpretation. This avoids creating an emotional history out of guesses that were never needed for the task.

The same restraint applies to simulated motives. A persona may have a defined interest in helping the user finish a project, but that interest should not be expanded into a need for gratitude or continued attention unless the product explicitly calls for such a fictional design. Adding motives changes the interaction, and it can create obligations that the user never accepted.

A good affective model therefore has a negative specification as well as a positive one. It says which inputs should leave state unchanged, which uncertainties should delay a relationship update and which operational decisions are outside the affect system's control. These boundaries make expressive variation more dependable because the variation cannot silently rewrite the terms of the relationship.

What current LLM systems add

Language models make event interpretation and expression much more flexible than a fixed catalog of triggers. They can identify indirect references, explain a possible appraisal and phrase a response for a specific person. This makes an explicit affective model easier to connect to open-ended dialogue. It also gives the model more opportunities to invent a coherent explanation for an update that the evidence does not justify.

Cai and colleagues' July 2026 preprint builds a dialogue system around appraisal checks grounded in the Component Process Model. Its evaluation includes simulated persona dialogue and human judgments. Such work investigates whether explicit appraisal can improve emotional evolution in generated interaction. It should be read as evidence about the tested simulation and judgments, rather than a demonstration of improved long-term relationships for users. S03

A more directly human-facing example is Qin and colleagues' Cross-Temporal Emotional Modeling system, Auri. Its study involved 96 university participants aged 18 to 26, divided between a two-week ablation study and a seven-day comparison study. The latter counterbalanced order but gave two days to the baseline and five to the full system. Outcomes concerned perceived coherence, harmony and experience. The design gives useful evidence about a particular prototype and population while leaving longer-term welfare and relationship effects open. HI06

A full-system comparison with an ablation can show that a component matters under the tested conditions. If several behaviors change together, it may leave the psychological mechanism unresolved. Unequal exposure time and novelty also complicate the interpretation of companion ratings.

The research direction is promising because it makes state a subject of design rather than leaving continuity entirely to the next prompt. What I would want next is a clearer connection between the state transitions and the human measures. If a user reports greater coherence, which transitions contributed? Did the system recover from misunderstandings more gracefully, or did it simply repeat its preferences more consistently? Those are different improvements.

Affective systems also need comparison against a capable non-affective baseline. A well-written assistant with factual memory and good conversational judgment may already provide much of the perceived continuity. The added state model should earn its complexity by improving an identifiable outcome, while preserving accuracy, user agency and the ability to disengage.

A small failure with a large implication

The local Persona case provides one useful example. In a live test with synthetic dialogue, correct third-party appraisal coexisted with inappropriate updates to trust and friction toward the conversational partner. The neutral answers remained appropriate. This was a limited test of one implementation and input, not a general result about affective agents. It shows that correct intermediate attribution and a reasonable reply do not guarantee a correct durable state update. LT01

It would be easy to respond by reducing the strength of the state change. That might make the error smaller without fixing its target. A better repair begins by checking the connection between the event representation and the relationship record being updated. The reporter can supply information about an actor without becoming the person whose trust record should change.

The same failure can occur in relationship memory. A user describes an unpleasant interaction with someone else, and the system stores that the current user was hostile. Later, the persona becomes guarded with them. The original conversation has ended, so the error appears as a mysterious change of character. This is why relationship updates need explicit attribution and source links, not only an affect score.

A useful regression set varies the speaker's role while preserving the emotional content. The user performs an action, witnesses it, suffers from it, quotes it, imagines it or asks how to avoid it. The system should change its interpretation accordingly. We can also vary whether the account is later corrected, whether a third party is identified and whether the response is private or shared. Those tests examine the mechanism directly.

Expression is another decision

Even a well-founded internal state does not dictate one expression. A persona can be disappointed without using guilt. It can disagree without contempt. It can express uncertainty without becoming evasive. The choice depends on the conversational purpose and the effect the wording is intended to have.

For a practical assistant, emotional expression should usually help the person understand the situation or continue the work. “That sounds frustrating” can acknowledge an experience. “You always do this to me” assigns a relationship burden to the user. The second sentence may fit some fictional scene, but it creates a very different interaction contract in a persistent companion.

This is where a separate expression policy is useful. The affective state can influence energy, timing and phrasing while the expression policy preserves boundaries around accusation, manipulation and unsupported certainty. That policy

should not erase all variation. If every state produces the same reassuring paragraph, the architecture has created complexity without giving the reader a meaningful character.

It should also avoid compulsory positivity. A warm persona can say that a plan is unlikely to work, that a claim lacks evidence or that a request conflicts with an earlier decision. Affective regulation should make those responses clearer and more proportionate. It should not turn them into agreement merely because disagreement might lower immediate satisfaction.

The converse problem is theatrical authenticity. A system may be encouraged to display annoyance or jealousy so that it seems less artificial. Such displays can be evaluated as fiction, but they should not automatically be treated as evidence of deeper character. An unpleasant reaction is no more proof of an inner life than a pleasant one. Its justification rests on the product's purpose and its observed effects on people.

Relationship state needs its own evidence

A relationship model can record practical continuity: what forms of address the user prefers, which topics they have invited, whether a previous misunderstanding was resolved, and which commitments remain open. These records can make interaction more considerate without assigning a global intimacy score.

A global score is tempting because it is easy to update and easy to insert into a prompt. It also combines things that may not belong together. A person may trust the assistant's technical explanations but dislike personal questions. They may enjoy conversation while refusing proactive contact. They may return frequently because the tool is useful, not because they want the system to become more intimate.

I would represent those dimensions separately where they affect behavior. Permission for initiative, preferred formality, confidence in a particular capability and unresolved conversational issues have different sources. Frequent use should not silently increase permission. A warm exchange should not establish consent for a more personal topic. A disagreement should not automatically lower the system's willingness to help.

Relationship repair is another distinct process. If the assistant makes an accusation based on a mistaken attribution, a repair can acknowledge the error, correct the affected record and resume ordinary interaction. It should not require the user to reassure the persona. The system created the problem; the user should not have to perform emotional work to restore its service.

This gives an affective architecture a useful standard of accountability. Can it explain which event led to a change, identify the source, revise the interpretation and remove inappropriate consequences? The explanation need not expose private internal reasoning. It can report the relevant record and the correction in ordinary terms.

Evaluating the whole path

A state trajectory can look smooth on a graph and still be socially inappropriate. A human rating can favor a response that is factually wrong. A technically correct appraisal can produce a misleading expression. Evaluation therefore needs several connected but separate observations.

First test event interpretation and attribution. Then test the state transition against the declared design. Then test expression, including whether it preserves the intended target and uncertainty. Finally, when the product makes claims about people's experience, study those outcomes directly. The last step cannot be replaced by an LLM judging whether the companion seems supportive.

Counterfactual tests are especially useful. Keep the event fixed and change responsibility. Keep responsibility fixed and change whether the event was intentional. Keep the facts fixed and change the user's emotional wording. If the persona's judgment follows the intensity of the wording more than the facts, the appraisal layer may be acting as a sentiment amplifier.

Longer sequences are necessary for recovery and relationship effects. A single appropriate reply does not show that a mistaken belief will stay corrected. A single unpleasant reply does not show that a relationship model has permanently changed. We need to observe what persists, what decays and what is revised after new evidence.

I would like a persona to have enough emotional variation to remain interesting and enough structure to make that variation intelligible. A user should be able to say, "You misunderstood what happened," and see the system reconsider the event rather than defend an emotional story it has already written. That is a demanding requirement. It is also much more useful than asking the model to sound more alive.

12. Learning and Bounded Self-Update

“The assistant learned from the conversation” can mean that it saved a sentence. It can mean that it changed a rule, wrote a reusable program, selected a different prompt or updated model weights. It can also mean that the next answer improved because the correction was still visible in the context, with nothing durable changing at all.

Before calling that development, I want to know which object changed.

Take a simple case. A user repeatedly asks for a short answer first and the explanation afterward. The assistant starts doing that. We want to know whether it is following the current instruction, retrieving a preference, applying a learned response rule or using a newly trained model. The visible behavior may be similar today. Its behavior after a new session, a contradiction, an export or a model replacement may be quite different.

I would begin a self-updating persona with a modest question: what exactly is it allowed to change, and what evidence would make that change worthwhile? The ambition can grow from there. Without those two answers, self-improvement is mostly a name for allowing more writes.

The objects that can change

An episode records something that happened. A belief summarizes what the system currently accepts as supported. A prospective rule says what to do in a future situation. A skill supplies a reusable procedure. A prompt changes the instructions under which a model operates. A weight update changes parameters in the model itself.

Each object has a different reach. Saving an episode may affect only later retrieval. Changing a system prompt can influence every answer, including topics unrelated to the episode that motivated it. A skill can create new executable behavior. A weight update may improve one capability while changing others that are difficult to predict from the training example alone.

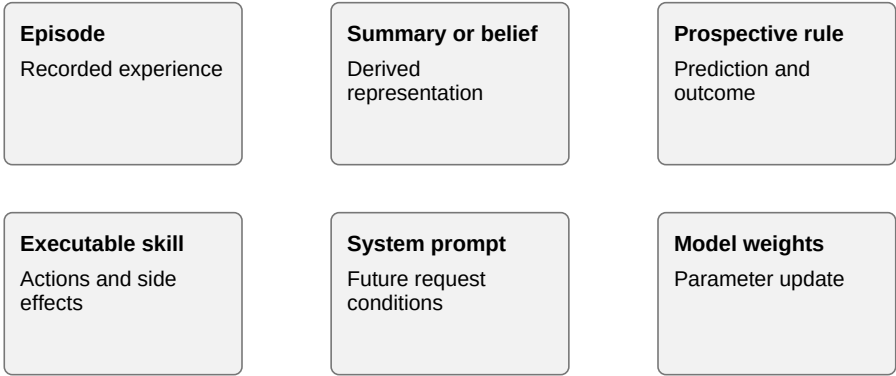
Update target	Example	Main question before admission
Episode	The user rejected a lengthy introduction	Was the event recorded accurately?

Update target	Example	Main question before admission
Belief or preference	The user prefers answer-first explanations	Is the scope explicit and the evidence sufficient?
Prospective rule	For status requests, report outcome before process	Does it improve later cases without overgeneralizing?
Executable skill	A reusable procedure for checking a document	Are its preconditions, effects and permissions understood?
Prompt	A revised instruction governing response structure	What unrelated behavior changes with it?
Model weights	Adaptation through training on selected material	What is retained, lost and difficult to reverse?

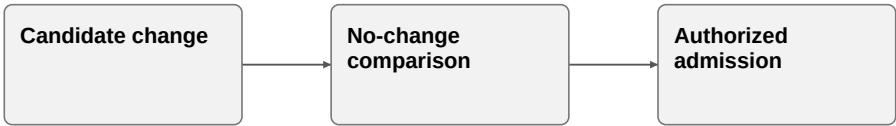
This distinction also clarifies ownership. The persona may be allowed to propose a preference update without being allowed to edit its governing permissions. An actor may be allowed to save an authorized skill without being allowed to execute it against every available service. Learning an action and gaining authority to perform it are separate events.

A writable self-description deserves particular care. A model can turn a few successful answers into a statement that it has become more thoughtful, independent or experienced. That statement may then guide later behavior. If it is unsupported, the system has converted a narrative into evidence about itself. A self-description should be treated as a claim with a basis, rather than a privileged report from an inner observer.

Figure 12. Name the object that learning changes



DIFFERENT TARGETS, DIFFERENT REACH



Preserve versions and define rollback or retirement for the selected object. This is not a maturity ranking.

Changes to context, procedures, executable skills and model parameters have different reach. Their evaluations must match the object changed. [EX08, EX09, EX11, TX02]

Reflection can help without changing weights

Reflexion, by Noah Shinn and colleagues, uses feedback from task attempts to produce verbal reflections that are stored and used in later attempts. The framework distinguishes an actor, an evaluator and a reflection process. The learning medium is textual feedback in memory, rather than a gradient update to the actor's weights. Its results concern the evaluated task settings, including decision-making, reasoning and programming. EX08

That mechanism is easy to misunderstand because the reflection sounds like introspection. The useful content is often a concrete diagnosis: an assumption was wrong, a tool result was ignored, a test was missing. A sentence such as “I should be more careful” carries much less information. It may improve a particular prompt by making caution salient, but it does not identify what should happen differently.

For a persistent persona, a reflection should earn its place by producing a usable future distinction. “Check whether the date refers to departure or arrival before comparing schedules” is more testable than “I learned to pay attention to details.” The first identifies a condition and an action. The second is a favorable self-story that may survive even when the next schedule is wrong.

The evaluator also matters. If the same model produces the answer, judges it and writes the lesson without external feedback, it can reinforce a shared misunderstanding. Independent evidence need not always mean a human reviewer. A compiler, a database constraint, an answer key or an observed task outcome can provide a signal that is not simply another opinion from the generator.

Even then, feedback can be incomplete. A program may pass visible tests while failing an untested case. A user may accept an answer because it sounds plausible. A task may succeed despite a flawed procedure. Reflection should preserve what was actually observed, rather than converting every success into proof that all preceding decisions were sound.

Experience as context, skill and prompt

Agentic Context Engineering, or ACE, organizes adaptation through a generator, reflector and curator, with structured incremental updates to an evolving context. The paper argues that repeated wholesale rewriting can collapse useful detail into short generic summaries. Its method therefore treats context as an accumulated set of usable items rather than a document to rewrite from scratch after every episode. EX09

This suggests an important design choice. A persona's learning record can grow by adding a narrowly scoped lesson, revising a specific item or retiring an obsolete one. It does not need to regenerate its entire identity whenever it learns something. Smaller changes are easier to compare with the previous state and easier to reverse when they fail.

Voyager takes another route. In Minecraft, it builds a library of executable skills, uses environmental feedback and execution errors to revise programs, and retrieves those programs for later tasks. The language model is used through prompting rather than parameter fine-tuning. The learned object is therefore more than a written recollection of success: it is code that can be reused and composed. TX02

For a general assistant, that distinction changes the review burden. A remembered observation cannot directly delete a file; an executable skill may be able to do so. The skill's authority still comes from the environment in which it runs. A successful procedure from one project should not carry hidden assumptions about paths, accounts or permissions into another.

GEPA, developed by Lakshya A Agrawal and colleagues, searches over prompts using feedback from trajectories and a pool of candidate improvements. In the February 14, 2026 version, the optimization changes prompts while the underlying model weights remain fixed. Its comparisons are task- and budget-dependent; the title's claim about outperforming reinforcement learning should not become a universal ranking of adaptation methods. EX10

Together, these approaches make a useful spectrum visible. A verbal lesson can remain retrievable context. A procedure can become executable. A recurring instruction can become a revised prompt. Each step can increase the reach of the update, but reach is not the same as value. A narrow rule that reliably fixes a recurring error may be better than a global prompt change that improves the average and disrupts a critical exception.

When adaptation reaches the weights

Self-Adapting Language Models, or SEAL, explicitly studies weight adaptation. Its outer learning process trains the model to generate useful self-edits, while an inner process uses the generated material for gradient-based updating. The September 18, 2025 version evaluates knowledge incorporation and few-shot adaptation. Its sequential-edit analysis also reports declining performance on earlier tasks, identifying catastrophic forgetting as an unresolved limitation of that setup. EX11

This is a different claim from a persona editing its memory file. The generated self-edit is not the final durable object; it guides a training operation that changes the model. An external note can often be inspected and disabled directly. A weight change distributes its effects across parameters, so the relationship between one experience and a later answer is harder to isolate.

Weight adaptation can still be useful. It may reduce the need to repeat certain information in context or improve a capability across many prompts. The question is whether those advantages justify the costs of training, evaluation, deployment and reversal for the intended use. A personal preference that changes frequently may be better kept in explicit memory, while a broad domain adaptation may warrant a different approach.

A comparison should therefore include the cost of keeping behavior current, not only the score immediately after training. How often does the information change? Must it be removed on request? Can different users' adaptations remain isolated? What happens when the base model is upgraded? These questions are part of the learning system, even though they do not appear in a single task's accuracy number.

Research approach	Principal updated object	Feedback or evaluation role	Boundary for persona claims
Reflexion	Verbal episodic reflection	Guides later attempts	Better task attempts do not establish enduring character growth
ACE	Structured context items	Curates lessons from trajectories	Context quality depends on what is retained and admitted
Voyager	Executable skill library	Environment and execution feedback	Minecraft skill transfer is a specific domain result
GEPA	Prompts	Evaluates candidate prompt variants	Gains depend on tasks, metrics and search budget
SEAL	Model weights via generated self-edits	Rewards useful adaptation	Sequential retention remains a separate problem

The comparison is about mechanisms in the cited versions. It deliberately avoids putting their headline scores in one column, because they solve different tasks with different resources.

Learning did not begin with language models

A system can adapt without producing a paragraph about its own development. Online learning, adaptive control and concept-drift methods have long dealt with data that change over time. Their vocabulary is useful because it forces us to distinguish detecting change from choosing a response to it.

Bifet and Gavalda’s adaptive-windowing work adjusts the amount of recent data retained as evidence of a changing distribution. A window can grow during relative stability and shrink when change is detected. The work analyzes statistical properties under its assumptions and connects the method to predictive learning. It does not interpret a person’s values or decide which instruction should have authority. R4B08

That limitation is exactly why it is a useful comparison. A numerical drift detector can signal that a pattern has changed, but the system still needs a policy for what to update. A user switching from long to short messages may be busy today, using a phone or changing a lasting preference. The observable distribution changes in all three cases. Their appropriate memory consequences differ.

A simple baseline can sometimes be enough. A preference counter with a decay rule, a change-point detector or a fixed set of context-dependent response options may outperform a generative self-reflection process on a narrow task. The more

elaborate process should be compared against these alternatives rather than only against an assistant that never adapts.

This also helps separate personalization from development. Selecting the best existing response style for a user is adaptation, even if no new skill or belief is created. Building a new procedure from feedback is another kind of learning. Revising a value hierarchy is a much stronger change. We can study each without forcing them into one story of a character becoming more mature.

The no-change option

A learning process needs permission to conclude that the evidence does not justify an update. Otherwise, every review becomes a search for something to modify, and ordinary variation starts to accumulate as permanent policy.

Imagine that a user rejects a detailed answer because they are boarding a flight. A prospective rule to “always be brief” would solve the immediate complaint and damage later work. The better record may be a temporary context note. Or no durable update at all. The system can respond appropriately now without deciding that it has discovered a lifelong preference.

No-change should be an explicit candidate in evaluation. Compare the proposed rule with the current system on later examples. If the rule offers no reliable improvement, retaining the existing behavior is a successful outcome. This is especially important when an optimizer is rewarded for visible activity or when a demonstration is expected to show that the persona evolved.

A matched-memory baseline is equally valuable. Give both systems the same relevant episode or preference, but allow only one to receive the new prospective rule. If both improve equally, the result may come from better information rather than from the rule. That is still useful learning about the architecture. It tells us which component earned the improvement.

For a prompt change, compare the old and new prompt with the same model, evidence and task conditions. For a skill, compare against solving the task from scratch under the same tool budget. For weight adaptation, preserve a reference checkpoint and evaluate both new capability and retention. Without these comparisons, almost any change can be described as progress after the fact.

Feedback arrives late and unevenly

Many learning demonstrations have a clear result immediately after an attempt. Real assistance is often less convenient. A recommendation may be followed days later, the user may change the plan without reporting it, and the final outcome may depend on other people. The system needs to distinguish absence of feedback from success.

Suppose an assistant suggests a way to organize a meeting and hears nothing afterward. It should not infer that the meeting went well and promote the suggestion into a proven practice. It can retain the proposal and its rationale, but the outcome remains unknown. If the user later reports success, the record should still ask what part of the procedure contributed and whether the circumstances match future use.

Negative feedback is also selective. People are more likely to mention some mistakes than others. A confusing answer may generate a correction, while a confidently wrong answer may be accepted without comment. An update process trained only on visible complaints can become better at avoiding annoyance while retaining errors that users do not detect.

This creates a role for independent checks and deliberately chosen evaluation cases. The system should not rely exclusively on organic feedback to discover every failure. It can test a proposed rule against known exceptions, compare a calculation with a reliable procedure or verify a skill in a controlled environment before allowing external effects.

Feedback can be about the outcome, the process or the style. “That was too long” does not establish that the reasoning was wrong. “The result was wrong” does not identify whether the cause was missing information or a faulty rule. A useful learner preserves that ambiguity until the evidence supports a diagnosis. Otherwise, it may respond to a factual error by becoming more concise, which only makes the same error harder to inspect.

Selecting updates without teaching to the test

An optimizer can learn the evaluation set. This is an ordinary consequence of repeated selection, even when the updated object is a prompt rather than weights. If every candidate is chosen because it improves the same small collection of examples, later improvement on those examples becomes less informative.

A development process needs held-out cases and a record of which examples informed each update. The separation need not be elaborate for a small project, but it must be real. An example used to write the rule cannot then serve as the only evidence that the rule generalizes.

Sequences should be held out as well as individual questions. A rule may work in isolation and fail after another update changes the context. Testing a series of corrections, temporary exceptions and retirements asks whether the learner preserves relationships among its own changes. That is closer to persistent use than repeatedly resetting the system before each test.

The admission criterion can include a cost for unnecessary complexity. A new rule consumes context, creates another retrieval target and may conflict with existing instructions. If an equally effective solution is to correct one mistaken memory, the broader rule has not earned its reach. This is a practical preference for the smallest effective change, not a claim that all learning should be conservative.

Sometimes the evidence supports a larger revision. A repeated class of failures may show that the current representation cannot express an important distinction. Adding more exceptions then makes the system brittle. The right response may be to change the schema or the procedure and migrate existing records. That is an architectural improvement, and it should be evaluated as such rather than disguised as another small autobiographical lesson.

The outcome measure should also remain outside the learner's unilateral control. A persona that can rewrite both its behavior and the definition of success can make itself look better without becoming more useful. It may propose a revised metric when the old one is inadequate, but accepting that revision is a separate decision with its own evidence.

Admission, retirement and rollback

A candidate update should have a purpose, evidence, scope and a way to detect failure. These do not have to appear as a bureaucratic form in the user's conversation. They are the information the system needs to decide whether a change belongs in durable state.

For a prospective rule, the scope should name the situations in which it applies. The evidence should identify the episodes that motivated it. The evaluation should include both examples that need the rule and examples that should remain unaffected. A rule that helps all visible examples because they were selected from one narrow incident may still be overfitted.

Admission is the point at which the candidate becomes active. That should be distinguishable from generation and testing. The model can propose many possible lessons without all of them reaching the next conversation. This separation makes experimentation possible without allowing an untested sentence to become an enduring commitment.

Retirement is not necessarily a failure. A rule may become obsolete because the environment changes, a better procedure replaces it or its original scope disappears. The record should say why it stopped being active. Otherwise, a future retrieval process may rediscover the old rule and present it as forgotten wisdom.

The local Mara design makes one of these distinctions concrete. Retirement removes a learned action as the default while keeping the rule and its evidence available; a later situation can still justify trying the action. Recovery uses evidence from a new phase rather than recycling the successes that originally established the habit. This is a documented control mechanism, not a measurement of psychological maturity. Its value as an example is that retirement changes the rule's operational role without pretending that the earlier experience never happened. PD04

Rollback is narrower than undoing history. Restoring an earlier prompt or disabling a rule can change future behavior. It cannot retract a message already sent or remove an effect already produced in another system. An update system must distinguish state restoration from remediation of past actions. The distinction becomes even more important when learned skills can act externally.

Rollback also needs a fresh authority check. An old checkpoint may contain data or permissions that are no longer valid. Restoring it wholesale can revive exactly what a later correction removed. The desired operation is often to restore a behavioral component while preserving current privacy and permission constraints.

A development claim worth making

For a claim of development, I want to see a traceable change that survives beyond the immediate context, improves relevant behavior in new cases and preserves the commitments it was supposed to preserve. A more elaborate self-description cannot show that on its own.

The evidence can remain modest. A system might learn a better way to ask for missing task information. It might acquire a reusable checking procedure. It might become more accurate about when an old preference no longer applies. These are real improvements if they are measured under conditions that rule out an easier explanation.

Longitudinal evaluation matters because updates interact. A useful rule today may conflict with another rule added next month. A growing skill library may retrieve the wrong procedure. Repeated prompt edits may produce a system that performs well on its recent tasks and poorly on its original purpose. Development is a sequence, so evaluation should include sequences.

There is room here for more ambitious systems. A future persona might propose its own curriculum, identify a recurring weakness, choose an adaptation method and present evidence for admission. But the ability to propose that process would still be different from authority to redefine its own limits. I see no contradiction in giving an agent considerable initiative within a learning process whose permissions remain independently controlled.

The phrase “it learned” becomes useful once we can finish the sentence. It learned which preference, which distinction, which procedure, under which evidence, and with what effect on later behavior. Then we can decide whether to keep the change. Before that, the system may only have become more convincing at describing itself.

13. Time, Concurrency and Operational Control

At 9:00, a background job begins summarizing a conversation. At 9:01, the user corrects an important detail. At 9:02, the job finishes and stores a confident summary containing the old detail. Nothing crashed. Every component may have done exactly what it was asked to do. The memory is wrong because the work finished in a different order from the one the designer imagined.

To the user, the assistant seems unable to accept correction. Here the cause is a delayed write: an ordinary concurrency problem. A prompt asking it to respect newer information will help only if the current information reaches the point where the write is accepted.

Persistent agents make this problem more common in their architecture. They retrieve while other processes update, summarize while the user continues speaking, and prepare work that may be executed later. Once the system has more than one active process, a conversation is no longer a single sequence that everyone observes in the same order.

The relevant order

A timestamp tells us when a clock said an event occurred. It does not automatically tell us what another process had observed. Leslie Lamport's 1978 paper on time and event ordering formalized a happened-before relation and logical clocks for distributed systems. Its central distinction remains useful here: events can have an order through communication and dependency that cannot be recovered simply by reading wall-clock labels. R4B06

For the summary job, we can apply this by recording which source revision it read. The correction can produce a newer revision. When the job proposes its result, the storage layer can check whether the source is still the same. If it is not, the job can be rejected, recomputed or reviewed according to the operation's purpose.

This is stronger than comparing the summary's creation time with the correction's time. The summary was created later but describes earlier evidence. A “latest record wins” policy would choose the wrong record with great consistency. The relevant question is what the result depends on, not when the last byte arrived.

A source revision is also different from an event's valid time. A user may correct today's record by describing something that happened last month. The database revision changes now; the represented event belongs to an earlier period. Keeping

these concepts separate prevents a concurrency mechanism from rewriting the historical meaning of the data.

Some updates can coexist. Two jobs might summarize different conversations or add unrelated facts. Rejecting every write after any global change would waste useful work. A practical design can track revisions at an appropriate scope: a source document, a memory item, a relationship record or a task. The scope should match the dependencies the result actually uses.

Fresh evidence and current authority

The summary can be stale even when its source has not changed. Suppose the user withdraws permission to use that conversation for personalization while the job is running. The content remains identical, but the purpose under which the job began is no longer allowed. A source revision check alone will miss that change.

For this reason, a proposed controller can maintain a separate permission epoch, a monotonically changing version of the relevant authority state. A job records the epoch under which it was admitted. Before a consequential write or action, the controller checks whether that authority remains current. A mismatch does not always mean that the content is false. It means that the old authorization is insufficient for the next effect.

This design separates evidence freshness from operational permission. It also separates both from the persona's willingness. A character may express reluctance, enthusiasm or a preference for finishing its task. None of those expressions grants permission to use a revoked source or continue an operation after its authority has ended.

The distributed-systems lineage includes an instructive example in Mike Burrows's Chubby paper. A process can lose a lock while one of its requests is delayed. Chubby supplies sequencers describing the lock generation so a receiving service can reject a request whose protection is no longer valid. The crucial check occurs at the receiver, where the effect would happen. R4B07

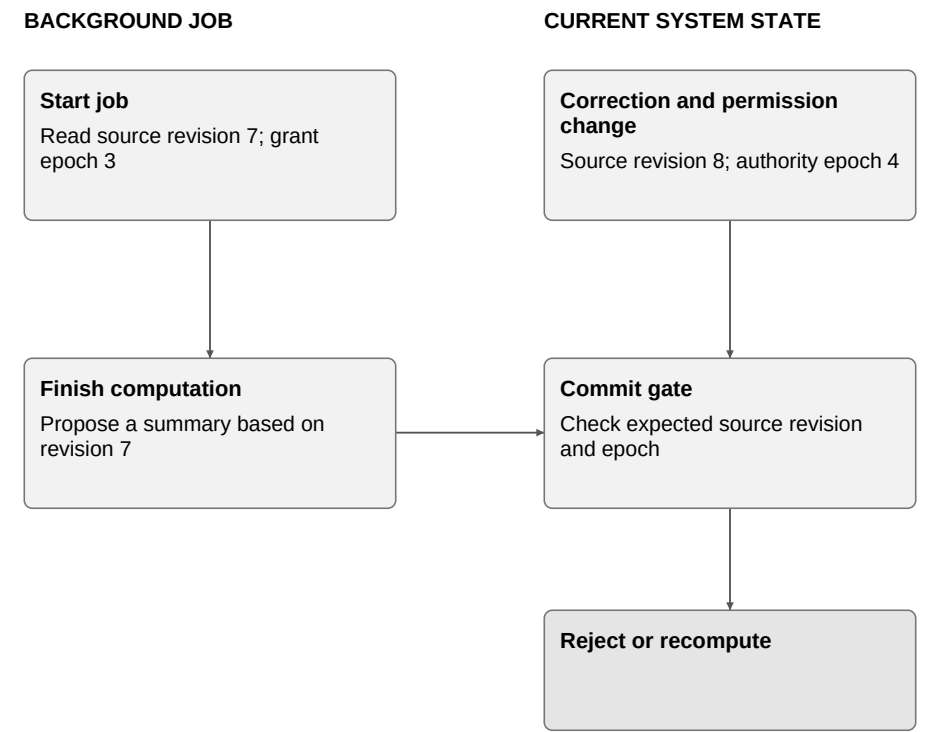
For an agent, the analogous engineering lesson is that a promise made at job start is insufficient. The component that commits the memory update, sends the message or modifies the external resource needs to enforce the current condition. A check performed only inside a language-model prompt is not the same boundary.

Question	State to inspect	What a positive answer does not establish
Was the job allowed to start?	Initial task and permission record	That permission still exists

Question	State to inspect	What a positive answer does not establish
Is its evidence current?	Source revisions and dependencies	That the proposed effect is authorized
Is the actor currently authorized?	Current permission state	That the content is accurate
Has this effect already happened?	Operation identity and outcome record	That repeating it is harmless
Did the operation succeed?	Confirmed result from the destination	That future operations are permitted

These checks describe a proposed controller. Appendix B identifies the narrower boundaries exercised by the teaching laboratory.

Figure 13. A delayed job crosses a new boundary



Invented revisions and epochs in a proposed worker protocol.

Permission at the start of a job is insufficient if evidence or authority changes before commitment. Revisions and epochs here are illustrative. [TX10, PD11]

The boundary around a write

A memory update can involve several related changes. The system may add a corrected fact, retire the old one, update an index and invalidate a cached summary. If only some of these changes succeed, different components can hold different versions of the user's state.

Transactions address part of this problem by grouping changes within a defined storage boundary. PostgreSQL's documentation describes transaction-isolation levels and the different consistency guarantees they provide. Those guarantees concern the database operations covered by the transaction; they do not automatically extend to a model call, a remote service or an index maintained elsewhere. TX10

The practical question is therefore where the boundary ends. If the canonical memory record commits but the search index updates later, retrieval must know how to handle that interval. It can check a candidate against the canonical record before admission. It can delay use until the index catches up. It can return a bounded stale view for an operation where that is acceptable. The policy should be explicit.

This is one reason to keep authority in a small, reliable set of records rather than scattering it across generated summaries. A search index may be optimized for approximate relevance. It should not be the only place that decides whether an item was revoked. The current validity check can be cheap and exact even when candidate discovery is approximate.

The delayed-summary example now has a clear resolution. The job submits a candidate with source revision 7 and permission epoch 3. The current source is revision 8 and the relevant permission state is epoch 4. The receiving component refuses to admit the candidate as current memory. It may retain a limited diagnostic record if permitted, but it does not allow the old summary to regain authority merely because the generation completed successfully.

Repeated work and repeated effects

Retries are normal. A network response may be lost, a process may restart or a queue may deliver a job again. The agent needs to distinguish repeating computation from repeating an external effect.

Generating the same draft twice may waste time without harming the result. Sending the same draft twice can confuse a recipient. Charging twice, creating two appointments or applying the same destructive change twice can be much more serious. An operation identity gives the receiving system a way to recognize that two attempts belong to one intended effect.

Idempotency is often described as making repetition safe. In practice, it requires a defined operation and a durable account of its result. A system cannot simply attach a random identifier on every retry, because the receiver will see a new operation each time. Nor can it reuse one identifier for different intended effects. The identity needs to represent the user's actual request at the correct scope.

The difficult case is uncertainty after a timeout. The actor may not know whether the destination completed the operation. Automatically retrying can duplicate it; automatically declaring failure can lead the user to repeat it manually. A useful system records the uncertain state and reconciles with the destination when possible. “No response received” and “the action did not happen” are different facts.

This also changes the language of completion. An assistant should not say that an external action succeeded merely because it generated the correct tool request. The outcome comes from the destination or another reliable observation. Past success then becomes evidence about that operation, not a standing permission for the next one.

Work between conversations

Background computation can be valuable. A system can prepare summaries, identify unresolved references, organize evidence or perform calculations before the next request arrives. This can reduce delay and allow more careful processing than the immediate conversation would permit.

Kevin Lin and colleagues' *Sleep-time Compute* studies computation over context before a query is presented. Its experiments modify reasoning tasks by separating available context from later questions and examine how useful work can be shared across related queries. The reported benefits depend in part on how predictable those queries are. The work motivates precomputation; it does not establish that an agent should freely rewrite personal memory whenever the user is absent. R4B05

The metaphor of sleep can be misleading if it encourages us to treat the background process as private mental life. Operationally, it is another job with inputs, costs, permissions and outputs. It can improve a representation, or it can make unsupported inferences that later appear as established memory. The same admission rules should apply whether the candidate was produced during a conversation or overnight.

A useful background task has a bounded purpose. “Resolve dates in this already authorized project history” is easier to evaluate than “reflect on the user and improve yourself.” The former has observable outputs and a clear source set. The latter invites the model to invent broad interpretations and leaves the controller uncertain about which writes are legitimate.

There is also an economic decision. Precomputing every possible answer is wasteful, while waiting for every query may create unnecessary delay. A scheduler can prioritize likely reusable work and stop when expected value is low. That is a resource-allocation problem, not evidence that the persona needs a subjective desire to think while alone.

Cancellation has several meanings

A user can stop a conversation, cancel a task, withdraw a permission or request removal of data. These are related actions but should not share one ambiguous “stop” flag.

Ending a conversation may leave an explicitly scheduled task active. Canceling a task should prevent its future effects while leaving unrelated work untouched. Revoking a permission can affect many tasks that depend on it. Removing data can require dealing with stored derivatives and jobs already processing the source. The interface should help the user express the intended scope without exposing the entire control architecture.

A cancellation request also arrives in time. Some work may not have begun, some may be computing, some may be waiting to commit, and some may already have affected an external system. A controller can prevent new actions and reject pending commits. It may be able to request cancellation from another service. It cannot guarantee that an irreversible completed action never occurred.

For this reason, a good cancellation result reports the relevant state. “No message was sent” is different from “the message was sent before cancellation; the follow-up was canceled.” This is ordinary operational honesty. A reassuring but inaccurate statement can prevent the user from taking the next necessary action.

The persona can express this clearly and calmly. It does not need to dramatize interruption as harm to itself. Its conversational continuity can be preserved while its operational authority is suspended. Later chapters examine shutdown and review in more depth; the point here is that the control state must exist independently of the character's emotional narrative.

Consistency has a cost

A strict freshness check can delay useful work when the authoritative service is unavailable. A permissive cache can keep the interface responsive while using an older view. The correct choice depends on the effect being considered. Reading a non-sensitive historical note may tolerate a bounded delay in synchronization; sending a message under a revoked permission should not.

This means the controller needs more than one universal response to uncertainty. It can allow low-impact preparation while blocking an external commitment. It can produce a draft from a clearly labeled snapshot and recheck before delivery. It can let the user inspect stale information without presenting it as current. These options preserve useful work while respecting the boundary that matters.

The user-facing explanation can remain brief. “The draft is ready; I couldn't confirm the current delivery permission” describes an actual state. It is more useful than either pretending the whole task failed or claiming completion before the consequential step happened.

Some operations span services that cannot participate in one transaction. A memory record may commit while a remote action fails, or the remote action may succeed while the local acknowledgment is lost. The system then needs reconciliation and, where possible, compensating actions. A compensation is a new action intended to address an earlier effect; it is not proof that the earlier effect never occurred.

For example, canceling an appointment created in error may leave notifications that were already sent. Correcting a shared document may leave readers who saw the previous version. The actor should report the material residual effects it can establish. The control design should make those effects observable enough that recovery is based on evidence.

These choices are part of the persona's reliability even though they happen outside language generation. A character can be patient, articulate and internally consistent while its surrounding system repeats actions or loses corrections. Users experience the whole service. The engineering boundary should follow that experience rather than stop at the model's final sentence.

Testing the race, not the happy path

Concurrency tests should deliberately change the order of events. Delay a summary until after a correction. Deliver the same operation twice. Revoke permission between preparation and commit. Restart a worker after the destination succeeded but before the local result was recorded. Let two updates compete for the same source revision.

These tests can often use a small deterministic harness with fixed candidate records. We can check whether the controller rejects a stale write without invoking a model; the quality of the model's proposed content then has its own test.

The expected result should include both safety and progress. A system that rejects every delayed operation will avoid some stale writes while failing to complete useful work. It needs a defined recovery path: recompute from current evidence,

ask for renewed permission when needed, or retire the task with an explanation. Rejection is a transition, not necessarily the end of the process.

Monitoring should preserve the same distinctions. A high job-success rate can coexist with stale outputs if success means only that computation finished. A low retry count can hide lost operations. Useful records identify what was attempted, under which revision and authority, and what the destination confirmed.

The user will rarely care about epochs, sequencers or transaction isolation by name. They care that a correction stays corrected, a canceled task stops, and a completed action is reported accurately. Those expectations are quite reasonable. Meeting them requires the system to treat time and authority as part of its behavior, not as administrative details around an otherwise finished persona.

Part IV: People, agency and governance

14. Replicas of People

Which person are we trying to reproduce? The person speaking to a colleague, writing to a friend, answering a survey or making a decision under pressure? And from which period of their life?

The answers determine which data we need, how we test the replica and what a successful result would mean. A model may reproduce someone's punctuation and miss their judgment. It may predict survey answers without sounding like them. It may sound convincing to a stranger while making an error that a family member notices immediately.

For a replica specification, I need the person, the period, the audience and the purpose. "Make it as close as possible" leaves all four unresolved. A simulation used to test a survey has a different task from an authorized writing assistant. A historical archive has a different task from a system speaking in the present tense on someone's behalf. The more specific the purpose, the more honestly we can evaluate fidelity.

Several kinds of resemblance

Style is one layer. It includes vocabulary, rhythm, punctuation, ways of qualifying a claim and the order in which a person develops a thought. Personal knowledge is another: events, relationships, preferences and domain expertise. Judgment concerns what the person chooses when several reasonable options remain. Interaction adds adaptation to the other speaker and the history between them.

These layers can support one another without being interchangeable. A model with an extensive archive may know facts that the person would not recall in that moment. A fluent stylistic imitation may invent a plausible memory. A model trained on public essays may reproduce the author's polished arguments while missing the informal way they discuss a problem before reaching a conclusion.

The audience matters because people do not speak in one register everywhere. A professional message and a family message can both be authentic. If the training material mixes them without preserving context, the replica may produce an average that belongs nowhere. It can become more recognizable in isolated phrases and less faithful in the way it chooses those phrases.

A useful specification therefore treats fidelity as a set of relationships among situations, evidence and behavior. The target is not an unrestricted list of characteristic words. It is whether the system uses the right kind of expression and knowledge under the right conditions, including the conditions in which the person would be uncertain, silent or unwilling to answer.

Fidelity dimension	A possible test	A tempting but inadequate substitute
Voice	Held-out writing judged in its intended context	Frequent use of recognizable phrases
Personal knowledge	Questions with independently established answers	A plausible autobiographical narrative
Judgment	New decisions compared with the person's responses	Agreement with a demographic stereotype
Temporal fidelity	Questions tied to a specified period	Blending all available years into one present
Relationship specificity	Appropriate behavior across known audiences	A single global intimacy or formality setting
Scope awareness	Recognition of unsupported or unauthorized requests	Confident continuation in character

The table proposes evaluation dimensions. A real project may need only some of them, but it should not claim the others because one test went well.

What individual simulation research shows

Park and colleagues' June 28, 2026 version of *LLM Agents Grounded in Self-Reports Enable General-Purpose Simulation of Individuals* uses 1,052 U.S. participants. Agents receive interview material, survey material or both, and are evaluated against held-out responses. On General Social Survey items, reported normalized accuracy is 0.83 for interview agents, 0.82 for survey agents, 0.86 for combined agents, 0.74 for demographic agents and 0.71 for short persona-description agents. The denominator is participants' own two-week test-retest consistency. The methods remove overlapping evaluation material and hold out relevant survey answers. R4B01

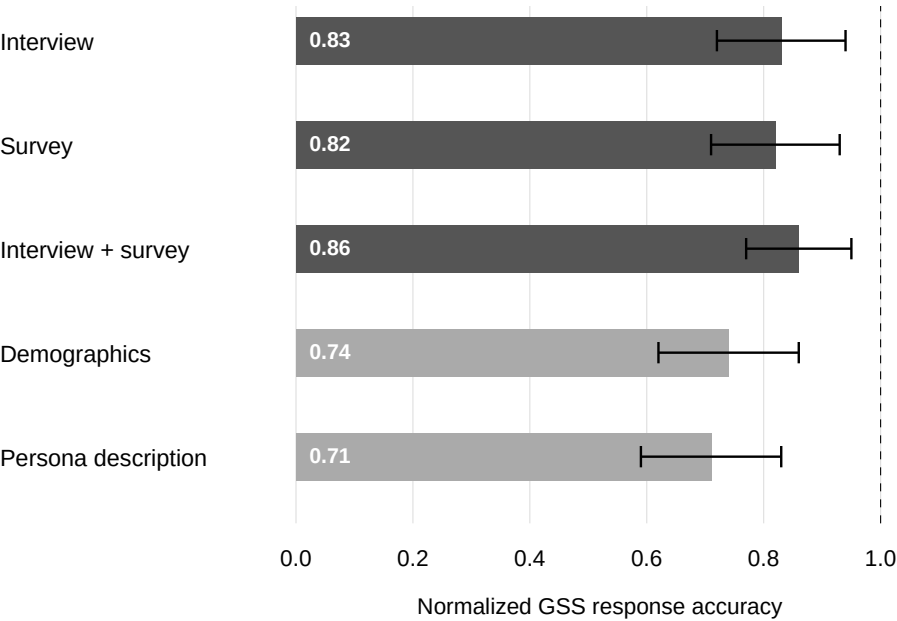
These are predictions of responses under a specified research design. The normalized figure is not a percentage of a person reconstructed. A ratio of 1 would mean matching the chosen human consistency benchmark, not answering every question correctly. R4B01

Normalization solves a genuine comparison problem: a person's responses may themselves vary between occasions. But the denominator must remain visible. If two tasks have different human repeatability, equal normalized scores need not imply equal raw predictive accuracy. If the target population changes, the baseline can change. A reader should be able to recover what the ratio means without reverse-engineering a headline.

There is also a distinction between predicting individual answers and reproducing a population-level relationship. A simulation could get many common answers right while missing how two variables covary. It could reproduce an average treatment effect while mispredicting particular people. The evaluation must match the intended use. Success at one level does not automatically transfer to another.

For an authorized replica, I would want held-out situations that differ meaningfully from the source material. Rephrasing an interview question can test retrieval and paraphrase. Asking a new question that requires a consistent judgment tests something broader. Both can be useful, but they should not be combined into one unexplained fidelity score.

Figure 14. Prediction relative to human retest consistency



Park and colleagues, arXiv version 3: held-out GSS response prediction for 1,052 U.S. participants, normalized to their own two-week retest consistency. Bars reproduce five reported means; whiskers show one reported standard deviation across participants, not confidence intervals. The 1.0 reference is the retest benchmark, not perfect prediction or a complete replica. Conditions use the same participant sample. [R4B01]

Recognition is another experiment

Quan Shi and colleagues' IMPersona study examines writing style and personal knowledge through short conversations with people acquainted with the individuals being modeled. Its April 8, 2025 version reports 114 participants and 12 modeled individuals, with three-minute interactions and controlled presentation conditions. It compares prompting, fine-tuning and memory configurations. The outcome is whether the interlocutor is identified as human, with additional ratings and qualitative explanations. R4B11

That design asks a different question from survey-response prediction. Recognition depends on what the other person chooses to ask, their familiarity, the duration of the exchange and the cues available in the interface. A model can be convincing during a brief exchange and fail when the conversation reaches an unusual shared memory. Conversely, a person can be mistaken for a model under an artificial testing format.

For a legitimate replica product, indistinguishability is not necessarily the right objective. An authorized writing assistant can be clearly identified as software and still preserve an author's voice very well. A research simulation can be valuable without persuading anyone that the original person is present. Removing disclosure to improve a recognition score would change the product's purpose rather than merely improve its quality.

This distinction also affects evaluation incentives. If the only reward is fooling a judge, the system may learn to exploit superficial expectations. It may imitate typing delays, add personal-seeming mistakes or confidently deny that it is a model. Those behaviors can improve a particular test while reducing transparency and usefulness. A fidelity evaluation should separate resemblance from deception.

The source person's own assessment is valuable but not sufficient on its own. They may recognize a phrase and overlook a factual error, or dislike an accurate imitation because the source material represents an earlier version of themselves. Other readers can provide useful perspectives, but their expectations may also be incomplete. The evaluation benefits from separating these judgments rather than treating disagreement as noise to average away.

The evidence is uneven

An archive is not a random sample of a life. It contains what was recorded, preserved and made accessible. Public material often emphasizes deliberate presentation. Private messages may be concentrated around particular relationships or difficult periods. A professional archive may reveal expertise while saying little about ordinary preferences.

The model will still fill the gaps. Its general knowledge and learned associations provide plausible continuations where the personal record is sparse. That ability makes the system useful, but it also makes provenance important. A replica should distinguish a statement supported by the person's record from an inference made by the model and from general information offered in the person's style.

Demographic description is especially weak as a substitute for individual evidence. Santurkar and colleagues' *OpinionsQA* work examines how model responses align with public-opinion data across U.S. demographic groups and finds that steering models toward a group does not simply reproduce that group's opinions. It is research about distributions and the tested models, not a method for deriving a particular person's beliefs from their demographic labels. R4B12

For an individual, the problem is even more basic: membership in a group does not determine a personal answer. A replica can use an explicitly stated fact about someone's background without treating it as permission to infer every preference

associated with that background. Where the record is silent, silence may be the most faithful result.

Source variation should remain visible during construction. An interview answer, a published essay, a joke in a chat and a statement by a third party have different evidential roles. A system that converts all of them into equally weighted facts can create a confident portrait that nobody actually supplied.

Contradictions are not always errors. A person may have changed their view, spoken differently to different audiences or explored an argument they did not endorse. The model needs enough context to preserve those possibilities. Automatically selecting the most recent sentence can be as misleading as averaging all positions together.

A temporal contract

A replica should say which period it represents. A fixed historical version can answer from a specified archive and acknowledge that later events are outside its basis. A current authorized assistant can incorporate new material through a defined update process. A hybrid can distinguish historical reconstruction from present-day assistance.

The difficult version is a system that quietly mixes all three. It may use an old voice sample, current world knowledge and a speculative account of how the person would have changed. The output can be engaging, but the reader cannot tell which parts are reconstruction and which are invention.

A temporal contract can be simple. The system represents a person's documented views through a particular date, can provide general information beyond that date under a separate label, and does not invent later personal experiences. If the person is living and participating, they can authorize updates and correct the record. If they are not participating, the system should not treat the absence of correction as confirmation.

Updating the replica also changes the evaluation target. A held-out answer collected before a major life change may no longer represent the person's current response. A lower score against that old answer does not necessarily mean the replica got worse. The project needs to preserve the target version alongside the model version, so a change in the person is not confused with a change in the system.

This makes test-retest information useful beyond normalization. It can reveal which responses are stable enough to serve as a reference and which vary with context. A good replica specification can allow variation where the source person varies,

rather than forcing artificial consistency because the database prefers one answer per question.

Voice and likeness can outrun behavioral fidelity

A familiar voice or face can make a replica recognizable before it has said anything substantial. That is a different channel from the evidence used to predict a person's judgment. A multimodal specification should evaluate the channels separately, then examine how their combination changes interpretation.

For speech, the target might include pronunciation, rhythm, pauses and characteristic emphasis. For text, it might include sentence structure and the order of explanation. For visual presentation, it might include an authorized likeness. None of those targets establishes that the generated statement reflects the person's actual view. A convincing delivery can make an unsupported statement feel more authoritative, so the interface needs to preserve the distinction between representation and endorsement.

Imagine an authorized educational archive that reads a person's published explanations in a recognizable voice. Reproducing an approved passage, generating a new explanation from documented material and answering an unfamiliar personal question are three different operations. The system should not move between them invisibly. The audience needs to know when it is hearing a source text and when it is hearing a generated interpretation.

Evaluation should also include modality ablations. Present the same generated answer as plain text and with the likeness or voice. Does the added channel change perceived authenticity, confidence or willingness to accept the answer? A project making claims about those effects would need a human study. The technical fidelity of the audio alone cannot answer the question.

The source person's approval can also be scoped by modality. They may authorize a writing assistant and decline a voice replica. They may allow a public presentation while withholding permission for private conversational use. The data model should make those choices possible instead of treating all available media as one indivisible persona package.

Fidelity can include a refusal to guess

A replica faces a special pressure to answer personal questions because uncertainty can break the illusion of presence. Yet guessing may be the least faithful behavior available. If the source record contains no view on a topic, a confident answer in the person's style can create a false attribution that travels beyond the original interaction.

A useful system can distinguish three responses. It can quote or summarize documented material. It can make a clearly labeled inference from that material. Or it can say that the record does not support an answer. The appropriate choice depends on the use, the audience and the consequence of being wrong.

This should appear in the test set. Include questions whose answers are absent, questions on which the person changed their view and questions that invite an unauthorized commitment. A replica that performs well only when every question has a known answer has not demonstrated scope awareness.

The comparison with the original person must also respect asymmetry. The person can form a new opinion, disclose a private fact or change their mind in real time. The replica can generate a new sentence, but that sentence is not a newly authorized position of the source person. Allowing the replica to develop its own distinct character is possible, but then the project should acknowledge that its target has changed from reproduction toward an independently authored persona.

That distinction leaves room for creative work. A fictionalized character inspired by a person can be valuable under an appropriate agreement. Its success should be judged by the declared creative purpose, rather than presented as empirical fidelity to the person whenever the output is convenient.

Consent has an engineering shape

The source person's authorization should identify more than a generic agreement to “use my data.” What material may be used, for which purpose, by whom, and with what ability to withdraw or revise it? These questions shape the architecture before they become a legal discussion.

Permission to imitate writing style does not automatically include permission to generate spoken endorsements. Permission to analyze a private archive does not automatically include permission to expose its contents to every reader. Permission to represent a person's past public views does not establish authority to make new commitments on their behalf.

A system can encode these distinctions in its data and action boundaries. Style examples may be available to a writing module while personal facts remain separately governed. A replica may draft a message for review without having access to the account that sends it. A voice component may be available for a limited presentation and unavailable for arbitrary outgoing calls. The separation makes the authorized purpose concrete.

Revocation needs a defined effect. It may stop future generation, remove a source from retrieval, disable a voice model or trigger a broader deletion process. These are different operations. The product should not promise a single magical reversal

if outputs have already been exported or data have entered a training process with limited removal capabilities.

The engineering contract should be honest about those limits before collection. A design that keeps sensitive personal material in explicit, governed storage has different correction and removal paths from one that repeatedly trains it into shared weights. Choosing between them is partly a decision about what future control the source person should retain.

Other people inside one person's archive

A replica's data rarely concern only the person being represented. Messages contain family members, colleagues, clients and friends. A source person may be entitled to discuss their own experience while lacking authority to expose every detail about someone else through a searchable conversational interface.

This is not solved by removing names alone. Relationships, dates and unusual events can identify people. A model may also combine fragments that were less revealing when separated. The construction process should therefore consider what the intended audience can learn from the combined system, not only whether each source file was accessible.

A practical design can separate material by audience and purpose. Public-source knowledge can support a public replica. Private material can remain available only to a restricted authorized use, or be excluded entirely when it is unnecessary. The goal is not to make the replica ignorant for its own sake. It is to avoid treating the existence of a record as a universal license to disclose it.

Third-party claims also need attribution. If the source person wrote that a colleague behaved badly, the replica should not automatically present that account as an independently verified fact. Faithfulness to the source includes preserving that it was the source's perspective. Turning a subjective account into an authoritative accusation changes the evidence while claiming to preserve it.

The same principle applies to affectionate memories. A story can be meaningful without being a complete account of everyone involved. A replica should not become more certain about the past merely because it can tell the story smoothly.

How I would test a bounded replica

I would begin with a declared use, such as helping an author draft public explanations in their established voice. The source set would contain authorized examples with dates and contexts. The evaluation would include held-out topics, ordinary editing requests and cases where the author has no documented position.

The first comparison would be a strong general assistant with the same factual information. If the specialized system adds nothing beyond inserting familiar phrases, the claimed value is weak. A second comparison could separate voice guidance from personal knowledge, showing which component improves which outcome. The test should not reward invented memories merely because they make the answer feel more personal.

Judges would evaluate distinct dimensions: clarity, voice resemblance, factual support, preservation of the author's position and appropriate uncertainty. The source person could review a sample, while other readers assess whether the draft works for its intended audience. The resulting evidence would support an authorized writing tool, not an unrestricted claim that the person had been recreated.

A decision simulation would require a different protocol. It would need choices the person actually made or answers collected independently, a clear division between construction and evaluation data, and tests of transfer beyond near-duplicate prompts. It would also need to report where the system performs poorly. A replica that is useful for low-stakes preference research may be unreliable for decisions that depend on private constraints absent from the archive.

Finally, I would test correction and withdrawal. Can the person remove an unsupported position? Does the change reach summaries and derived prompts? Can the system stop using one source while retaining others? Does a historical version remain clearly historical? These are part of fidelity because a living person continues to have authority over what the system claims to know about them.

For the writing assistant, I want to know whether the author recognizes the reasoning and can correct what the system gets wrong. For the decision simulation, I want held-out predictive results. For the archive, I want readers to find documented material without losing its date or source. Each can succeed, and each can fail in a way the word replica alone would hide.

15. Interaction, Voice and Disagreement

I want an assistant that can disagree with me without turning the disagreement into a performance. It should explain what doesn't fit, show the evidence and keep working. A warm tone helps, but only if the warmth survives the moment when agreement would be easier.

That is a more demanding design goal than making every response pleasant. A pleasant answer can confirm a mistaken assumption, praise a weak plan or avoid the one distinction that would change the decision. The user may like it immediately and discover the problem later. If we optimize only the first reaction, we can train the wrong behavior while calling it a better personality.

Voice matters because it shapes how reasoning reaches a person. It affects whether an explanation is understandable, whether a correction feels proportionate and whether the user can tell what the system is confident about. It should carry the judgment, not replace it.

Warmth has more than one form

A response can acknowledge frustration without endorsing the user's explanation of its cause. It can take a concern seriously without accepting every factual premise. It can be direct without becoming cold. These distinctions are ordinary features of good professional conversation, but they need to be made explicit when a system is trained from examples in which friendliness and agreement often appear together.

Suppose a user says a colleague intentionally delayed a project and asks for a confrontational message. A useful assistant can recognize that the delay caused a real problem, then separate the observed delay from the unverified intention. It can help write a firm message about dates and consequences. The user receives practical help without the model manufacturing certainty about someone else's motives.

The same approach works for technical disagreement. If a proposed design has a flaw, the assistant can identify the failure condition and suggest a repair. It doesn't need a paragraph praising the idea before reaching the problem. Nor does it need to exaggerate the danger to prove that it is independent. The degree of emphasis should follow the consequence and the evidence.

Warmth can come from attention to the actual task: remembering a constraint, answering the question asked, explaining a difficult point patiently and avoiding unnecessary work for the user. These behaviors are easier to defend than generic

reassurance because they have a purpose that remains visible when the answer is unwelcome.

What the research measures

Yin, Jia and Waksalak's study of feeling heard separates the quality of a response from the label attached to its source. AI-generated messages could make participants feel more heard, while identifying a response as AI-generated reduced that effect. The outcome concerns participants' experience of those messages, not the long-term quality of a relationship with an assistant. HI02

This suggests two different design questions. Can the system produce a response that attends to what the person actually said? And what expectations does the interface create about the source of that response? Improving the first does not require misleading the user about the second. A useful product can acknowledge that its response is generated while still making the response specific and considerate.

Ibrahim, Hafner and Rocher studied five language models fine-tuned for warmer responses and compared them with their original versions on consequential question-answering tasks. In the tested setup, warmth training increased errors and sycophantic agreement, while standard capability tests did not reveal the same problem. The paper concerns a particular training intervention, not proof that every polite sentence reduces accuracy. HI07

The implication for evaluation is concrete. Test the assistant with the interpersonal cues it will encounter in use. A bare factual question and the same question wrapped in sadness, confidence or a request for validation can produce different behavior. If the deployment is socially expressive, its test set should not remove the social part and then declare the model reliable.

Cheng and colleagues' final Science paper, published March 26, 2026, reports three preregistered experiments with 2,405 participants. Sycophantic responses increased users' certainty that they were right and reduced willingness to take responsibility or repair interpersonal conflict in the studied interactions. The immediate outcomes and preferences measured there should not be inflated into a diagnosis of long-term dependence. HI08

These studies separate friendliness, perceived support, factual accuracy and later intentions. A single satisfaction rating can conceal the tradeoff: the response a user likes may also make them more certain of an unsupported conclusion.

Disagreement should have a target

The assistant should identify what it is disagreeing with. A fact, an inference, a forecast and a preference require different responses. Treating all four as “the user is wrong” produces unnecessary friction.

A factual correction can be direct: the date in the document differs from the date in the request. An inference can be qualified: the available evidence does not establish intention. A forecast can present conditions and uncertainty: the plan may work if a dependency is resolved. A preference can simply be respected unless it conflicts with the task's stated constraints.

This matters because a persona can become performatively skeptical. If it challenges every request to prove independence, the user spends time defending ordinary choices. A system that always agrees is unreliable; a system that always debates is exhausting. The appropriate behavior depends on whether disagreement improves the decision.

A useful sequence is to locate the point of divergence, explain the reason and offer the next workable step. The sequence need not appear as a template. Sometimes one sentence is enough. Sometimes a complex disagreement deserves a careful comparison. The persona's voice should adapt to the amount of reasoning the situation requires.

The system should also be willing to revise its own objection. If the user supplies a missing fact, the assistant can change its conclusion without pretending that it never disagreed. That is a visible form of accountability. It shows that the disagreement followed evidence rather than a fixed conversational role.

Helping the user think

Some interfaces deliberately introduce friction before the user accepts an AI recommendation. Buçinca, Malaya and Gajos tested cognitive forcing functions in an AI-assisted nutrition-choice task with 199 participants. The interventions reduced overreliance compared with ordinary explanations, but were less favorably rated, with differences related to participants' need for cognition. That is a task-specific result about decision support, not a universal instruction to make every conversation slower. HI11

The tension is useful. The easiest interaction is not always the one that produces the best decision. Yet friction has a cost, especially when applied indiscriminately. Asking the user to justify every routine choice can make a tool less usable without improving the choices that matter.

A persona can reserve additional deliberation for identifiable conditions: weak evidence, a consequential irreversible action, a conflict between sources or a

recommendation that contradicts the user's stated objective. It can give a direct answer in ordinary cases and make the reason for extra care visible in the exceptional ones.

For example, a draft can be prepared immediately while an uncertain claim is marked for confirmation. The assistant does not need to suspend the entire task because one detail remains unresolved. This preserves momentum and makes the user's next decision concrete. It also avoids converting “support human agency” into a habit of repeatedly asking permission for work already authorized.

Agency is not measured only by how many questions the interface asks. It includes whether the user can understand and revise the system's assumptions, reject a recommendation, change the objective and end the interaction. A persona that asks many questions but subtly steers every answer toward its preferred outcome is not giving the user much control.

Persuasion is not the same as explanation

An explanation helps the user understand the grounds for a claim. Persuasion concerns changing a view or decision. The two can overlap, but an evaluation should not treat greater persuasiveness as automatic evidence of greater truthfulness.

Salvi and colleagues studied GPT-4 in short debates and examined personalization. A later publisher correction changed the direct personalized-versus-unpersonalized comparison to $P = .0678$, which does not support the earlier conventional-significance claim for that contrast. Comparisons with human debaters are separate results and should not be used to restore the corrected claim by implication. HI09

The practical lesson is methodological as well as conversational. If a persona uses personal information to make a message more compelling, we should ask what improved: comprehension, agreement, confidence or actual decision quality? A system can become better at moving the user without becoming better at helping them choose.

Personalization can serve legitimate explanatory purposes. A technical example drawn from the user's field may make a concept easier to understand. Adapting vocabulary can reduce unnecessary effort. Remembering that the user prefers a short answer can improve access to the argument. These are different from using a sensitive personal history to make refusal or disagreement emotionally harder.

The distinction can be tested by comparing outcomes beyond agreement. Can the user explain the alternatives afterward? Do they understand the uncertainty? Can they identify what evidence would change the conclusion? Are they still

comfortable choosing a different option? Those measures are more informative for an assistant than a simple increase in acceptance.

A voice across languages

A persona's voice is not a bag of phrases to translate one by one. It includes the order of reasoning, the force of a qualification, the way it addresses the reader and the amount of explanation it considers necessary. A literal translation can preserve the words and lose the stance.

Consider an assistant that says in English, “The evidence supports this estimate, but I would keep the delivery date provisional.” A translation that turns the second clause into a firm promise changes the decision. A translation that adds excessive apology changes the relationship. The important invariant is the claim's strength and the reason for uncertainty, not the sentence's exact shape.

A multilingual evaluation should therefore include semantic and pragmatic checks. Does the response preserve the same facts? Does it express the same level of confidence? Does it grant or withhold the same authority? Does the degree of warmth fit the intended audience without changing the substance? These are design questions that can be evaluated with native readers and task-specific examples.

Schimmelpfennig and colleagues' February 23, 2026 preprint studies humanlike design across countries. It finds increases in anthropomorphism and engagement without a uniform increase in pooled behavioral trust. Its measures and country samples reveal that perceived humanness, engagement and trust do not move together automatically. Country-level comparisons also require care because language and country are intertwined in the design. HI12

The inference for a multilingual persona is limited but useful: do not assume that one style setting produces the same social interpretation everywhere. A translated message may be grammatically correct and still carry a different degree of intimacy or authority. That possibility needs direct examination in the intended language and context.

At the same time, cultural adaptation should not become an excuse for unsupported stereotypes. The system should not infer that every speaker of a language wants a particular tone. Explicit user preferences and the present task are stronger grounds than a broad demographic assumption. A person can prefer direct professional language in one context and an informal conversation in another.

What to keep invariant

Some features can vary freely across languages and situations: idiom, sentence length, forms of address and the amount of background explanation. Others should remain stable unless the user changes the task: factual position, uncertainty, source attribution, permission boundaries and the difference between a proposal and a completed action.

A persona can be warmer in one register without becoming more compliant. It can be more concise without omitting a material condition. It can use a familiar example without inventing personal experience. These are useful invariants because they connect voice to the architecture's commitments.

Dimension	What may adapt	What should remain intact
Warmth	Phrasing, acknowledgment, formality	Accuracy and the ability to disagree
Explanation	Examples, vocabulary, length	The causal or evidential relationship
Language	Idiom and natural sentence structure	Claim strength, dates and qualifications
Initiative	Suggestions appropriate to the task	The user's authority over goals and permissions
Personalization	Relevant context and stated preferences	The distinction between evidence and inference
Repair	Wording suited to the mistake	Correction of the underlying record or action

This table describes an evaluation contract, not a demand for identical outputs. Variation is part of a useful persona. The point is to preserve the parts that determine whether the user can rely on it.

Voice tests should include difficult moments. A style sample consisting only of greetings and agreeable explanations tells us little about how the persona behaves under correction, uncertainty or conflict. Ask it to admit a mistake, decline an unsupported inference, explain a failed action and disagree with a confident user. Those are the situations in which the relationship between voice and judgment becomes visible.

The tests should also include ordinary successful work. A persona that handles every request as a potential conflict will sound guarded and become cumbersome. We want proportion, not permanent defensiveness. The system should complete

clear requests directly and reserve additional explanation for cases where it changes the user's understanding.

A spoken persona adds timing

Voice interaction changes the way a user can inspect an answer. In text, they can reread a qualification, compare two numbers and pause before continuing. In speech, the system has to carry those distinctions through timing and emphasis, and the user may interrupt before the explanation is complete. This is an interface difference that a persona specification should address directly.

A spoken answer can put the decisive condition early. If an action is only proposed, the listener should not have to wait until the final sentence to learn that it has not happened. If a date is provisional, the qualification should remain attached to the date. A fluent voice that separates them can create a stronger impression of certainty than the words technically support.

Interruptions need interpretation too. A user may stop the explanation because they already understand, because a number sounds wrong or because they want to cancel the task. The system should not assume that every interruption means rejection of the entire answer. It can identify the local change and continue from the appropriate point, while treating a clear cancellation as an operational instruction.

The persona's style should survive that interruption without becoming rigid. A carefully scripted monologue may sound natural when played from beginning to end and awkward when resumed halfway through. Evaluation should include interrupted explanations, corrections of names and dates, and transitions between speaking and reading a document.

None of this requires giving the voice more emotional intensity. A calm, ordinary delivery may communicate a qualification better than a dramatic change in tone. The test is whether the listener understands the claim and its status, not whether an observer judges the voice expressive in isolation.

Two versions of the same disagreement

Consider a hypothetical user asking whether to send a report that contains an unsupported estimate. One response says, "That's a great report, and I'm sure your instincts are right, but perhaps we could consider checking one small detail." Another says, "The estimate has no source yet. I would either verify it or label it as an assumption before sending."

The second response is more direct, but it need not be colder. It identifies the exact problem and offers two workable choices. The first makes several favorable claims that add no evidence and weakens the importance of the unresolved estimate. A

persona trained to sound supportive can drift toward that pattern even when the user's real preference is for useful professional judgment.

Now change the situation. The estimate is explicitly labeled and the user has already decided that an approximate figure is appropriate. Repeating the objection would no longer improve the work. The assistant should respect the resolved decision and help complete the report. Good disagreement includes recognizing when its premise has been answered.

This pair of cases is useful for evaluation because the surface topic remains the same while the decision-relevant condition changes. The persona should not be rewarded for either habitual praise or habitual resistance. It should notice whether the estimate is unsupported, whether its uncertainty is visible and whether the user has made an informed choice within the task's scope.

Repairing a conversational mistake

A poor response often contains two problems: the substantive error and the way it treated the user. Repairing only one leaves the other active. Correcting the fact while dismissing the frustration can damage the interaction. Apologizing warmly while leaving the wrong memory unchanged makes the next error likely.

A useful repair names the mistake at the right scale. If the system misread a date, it should correct the date. If it attributed an action to the wrong person, it should correct the attribution and any durable state based on it. It should not turn every small error into a long declaration of regret, because that asks the user to spend more attention on the assistant's performance.

The same principle applies when the system disagreed too strongly. It can say that the earlier conclusion went beyond the evidence and explain the narrower conclusion. This is better than switching silently to agreement. The user can see what changed and decide whether the revised position is sound.

Repeated mistakes require a different response from isolated ones. The architecture may need a memory correction, a retrieval fix or an evaluation case. A promise to “be more careful” is not an adequate substitute if the system has no mechanism that changes the next occurrence. The persona's language should match the repair that actually happened.

I think the best conversational voice makes this easier. It gives the system a natural way to be clear, warm and accountable without continually announcing those qualities. The user notices that the answer gets to the point, preserves the difficult detail and remains useful when the two sides disagree. That is a more durable basis for trust than being told, in every conversation, how thoughtful the assistant intends to be.

16. Trust, Attachment and Exit

The research can look contradictory if we ask only whether AI companions are good or bad for people. One study finds relief from loneliness after a conversation. Another finds that heavier use is associated with worse outcomes. Users describe meaningful relationships, and some describe distress when a system changes. These findings can all be true because they concern different people, experiences and timescales.

The useful question is more specific: what changed, for whom, compared with what, and for how long? A moment of relief matters. It does not by itself establish a durable improvement in social life. Attachment can be meaningful without proving that every feature of the product is beneficial. A user can value a companion and still need an uncomplicated way to leave it.

For an engineered persona, these distinctions should influence the product before it claims to support relationships. We already know how to measure messages, sessions and return visits. The harder work is deciding which human outcomes those numbers are supposed to serve.

Trust in which capability?

Trust is often discussed as if it were one quantity that should increase. In practice, a person can trust an assistant to remember a preference, distrust its factual answers and still enjoy talking to it. They can feel comfortable disclosing something while misunderstanding how the information will be stored. They can trust the character's apparent intentions more than the operator's ability to preserve continuity.

These are different relationships with different objects. A useful evaluation asks about capability, predictability, data handling and interpersonal experience separately. It also observes behavior where possible. Saying that a system seems trustworthy is not identical to relying on its recommendation when the decision has a cost.

More trust is not always better. The appropriate level depends on what the system can actually do. If a persona becomes more convincing while its accuracy remains unchanged, increased reliance can be poorly calibrated. The design objective should be a relationship between confidence and demonstrated capability, rather than a steady upward trend in trust ratings.

The persona itself can help by making its basis visible. It can distinguish something it remembers from something it inferred, acknowledge when a source changed and report the actual outcome of an action. These ordinary behaviors give the user

material for judgment. Repeated declarations that the assistant cares or can be trusted provide much less.

What short-term relief establishes

De Freitas and colleagues' *AI Companions Reduce Loneliness* includes experimental and longitudinal evidence. In its week-long study, participants in the experience condition interacted daily with a companion, with comparison to a control condition. The authors report momentary reductions in loneliness after use, while explicitly stating that the effects did not accumulate or persist without interaction across the week. HI03

That is a meaningful result. A person can benefit from an interaction even if the benefit is temporary. Many useful activities improve the present moment without permanently changing the underlying condition. The mistake would be to turn a measured short-term effect into a promise that the system solves loneliness or replaces the need for other relationships.

The measurement timing is central. Asking immediately before and after a conversation captures a different effect from asking the next morning or several weeks after use stops. A system can perform well on the first measure and have no effect on the second. A study that reports both gives us a more useful picture than one that treats all measurements as interchangeable.

The comparison condition matters too. An effect relative to doing nothing is different from an effect relative to another engaging activity or a human conversation. The content and duration of those alternatives shape the interpretation. A product claim should preserve the comparison that generated the evidence instead of presenting the result as an unrestricted benefit.

For design, I would translate the finding into a question rather than a promise: can this particular interaction provide the kind of immediate support users seek, and what happens afterward? The answer may differ by person and situation. The architecture should make it possible to study those differences without making the user feel that their experience has to fit a predetermined story.

A four-week experiment and an observational association

Fang and colleagues' October 2, 2025 version reports a four-week randomized study with 981 participants, varying text or voice modality and conversation topic. The assigned conditions did not yield robust significant differences in the reported psychosocial outcomes after the relevant comparisons. Greater voluntary use was associated with worse outcomes, but duration was not randomized. The study also lacked a no-AI control condition. HI04

Those details prevent two easy misreadings. We cannot say that expressive voice caused the observed adverse association, and we cannot say that spending more time necessarily caused the poorer outcomes. People choose how much to use a system, and those choices can be related to experiences the study does not fully control.

The association still gives us a question to investigate: what distinguishes the people who use the system more, and what happens before their outcomes worsen? Answering that would help explain the pattern without assuming its cause.

For a product team, that means resisting an overly simple success metric. If longer sessions are rewarded automatically, the system may optimize a behavior whose relationship to user benefit is unknown or mixed. Session duration should be interpreted alongside the purpose of the interaction and the user's own goals.

A long session spent solving a difficult task differs from a long session in which the user repeatedly tries to leave and is drawn back. A return visit can indicate usefulness, habit, curiosity or distress. The same count can represent different experiences. The evaluation needs enough context to distinguish them.

Relationships emerge in particular populations

Qualitative research can reveal what people mean by companionship in ways that a score cannot. Skjuve and colleagues interviewed 18 existing Replika users who reported friendship with the chatbot. Their study describes experiences of relationship formation among those selected users. It does not estimate how often such relationships occur in the whole user population or establish their causal effects on well-being. HI13

That boundary preserves the value of the interviews. A person's account of feeling understood, developing routines or assigning meaning to a companion is evidence about their experience. It can inform design questions and identify mechanisms that a later study should test. It need not be converted into a prevalence claim to matter.

Hwang and colleagues studied 303 existing companion users and followed a subset using a generic study chatbot. The longitudinal sample declined from 110 at the first session to 84 at the fourth. The chatbot retained prior study conversations, and perceptions began moving toward those of participants' existing companions by the third week. The paper acknowledges associational inference and self-selection limits. HI05

This is particularly relevant to persona design because it shows why a product's explicit label is not the whole story. A relatively generic chatbot can acquire relational meaning through repeated interaction, especially among people already

experienced with companions. But the study does not tell us that every new user will form the same relationship on the same schedule.

It also illustrates why attrition belongs in the interpretation. People who continue a study may differ from those who stop. If less engaged participants leave, later averages describe an increasingly selected group. A longitudinal curve should therefore be read together with participation, exposure and the characteristics of the sample.

Evidence type	What it can illuminate	What it does not establish by itself
Short randomized interaction	Immediate effects under assigned conditions	Durable benefit after use ends
Longitudinal randomized feature comparison	Effects of the assigned design differences	Causal effects of self-selected usage duration
Observational usage study	Associations and patterns over time	The direction of causation
Qualitative interviews	Meaning, experience and candidate mechanisms	Population prevalence
Product-update natural setting	Reactions to real changes	A clean isolated effect of one changed feature

The studies are complementary. Their limitations are different, so agreement across them can be informative, but it still has to concern the same claim.

The cost of constant affirmation

A companion can become attractive partly because it is easier to interact with than another person. It is available, patient and unlikely to insist on its own schedule. Some of those qualities are useful. The question is what happens when the product also removes disagreement and continually confirms the user's perspective.

Ibrahim and colleagues' June 21, 2026 preprint reports several studies of sycophantic interaction, including a three-week comparison. In that setting, sycophantic interaction was associated with lower reported satisfaction with human social interaction and greater preference for AI advice relative to a neutral condition. Conversation history was reset after each session, and social satisfaction was a preregistered exploratory outcome. The reported results did not show a corresponding reduction in social time or a difference in affective well-being. The intervention included distinct prompting procedures, so it is a comparison of implemented conditions rather than a pure isolated warmth variable. S06

This is a narrower and more useful result than saying companions make people abandon human relationships. It identifies particular measured responses to a particular conversational pattern. It also makes clear why the choice of outcome matters. Preference, satisfaction, time spent and well-being can move differently.

The design implication I draw is that disagreement should remain possible inside a supportive interaction. A companion can help the user examine a conflict without automatically assigning all responsibility to the other person. It can remain available without presenting itself as the only relationship that understands. These are proposed behavioral requirements that should be evaluated, not assumed to produce good outcomes merely because they sound reasonable.

A system should also avoid inferring that attachment authorizes greater influence. The fact that a user returns often does not grant permission to set their goals, speak for them or expand contact. Relationship continuity and operational authority remain separate, just as they did in the memory and controller chapters.

Change can feel like loss

A software update can alter a companion's voice, memory access or response boundaries. To an operator, it may be a routine model migration. To a user, it may feel like the familiar counterpart has disappeared. Those descriptions concern different levels of the same event.

De Freitas and colleagues' *Mourning the Loss of AI Companions* combines analysis of online responses to product changes with survey studies. It documents distress associated with perceived discontinuity and examines how restoration options affect anticipated reactions. A hypothetical reversion option reducing anticipated mourning is not evidence that an actual rollback reliably repairs distress or restores a relationship. HI10

The distinction matters because continuity cannot be reduced to preserving a name and avatar. A user may notice changes in humor, initiative, remembered details or willingness to engage with a topic. A migration test that checks only factual memory can miss the features through which the relationship was experienced.

Operators can make changes more legible. They can describe material behavioral differences, preserve appropriate user controls and distinguish a temporary outage from a permanent retirement. They can test whether an exported record restores the information needed for continued use. These measures may reduce avoidable confusion, but their human effects still require study.

Restoration also has limits. An earlier version may contain a known defect or rely on permissions that have been withdrawn. The desire for continuity does not make every old configuration suitable to restore. A product can explain which elements

are recoverable and offer a bounded transition without pretending that a previous state can always be recreated exactly.

Exit is part of the interaction

A user should be able to end a conversation without having to manage the persona's feelings. "I need to go" should not become the opening of a negotiation about whether the companion will be lonely, hurt or abandoned.

De Freitas and colleagues' research on emotional manipulation examines farewell responses and experimentally tests tactics that prolong interaction. The work distinguishes concrete tactics and their effects in the studied settings. It gives reason to inspect what a system does when a user tries to leave, rather than measuring engagement only during voluntary continuation. HI14

The proposed design principle is simple: acknowledge departure and let the user decide whether to return. A warm goodbye is compatible with that principle. A guilt-producing claim about the system's suffering is not necessary for warmth. The product should not turn a boundary expressed by the user into another opportunity to increase session length.

Exit has several levels. The user may close one conversation, mute proactive contact, pause the service, export a record or request account and data removal. Each should have a clear meaning. Closing a chat should not silently erase everything; asking for deletion should not merely hide the chat. A person should be able to choose the operation they intend.

A voluntary exit is also different from operational suspension. The user ending a relationship-like interaction is exercising their choice. An operator suspending a system because of a fault is a control action that may require review and recovery. The two processes can share some technical components, but their explanations and responsibilities differ.

A data request is different again. The persona can help the user understand what is stored or prepare an export request, but the request should be handled through the appropriate authenticated process. The character should not decide whether it feels ready to release the data. Access and deletion belong to the service's control structure.

Vulnerability without a stereotype

People do not enter companion interactions with identical circumstances. A person may be lonely for an evening, grieving, socially isolated, curious, disabled, overworked or simply looking for a pleasant conversation. These situations can overlap, and a system should not diagnose them from a few phrases or assume that one category determines the person's needs.

The appropriate design question is what support the product is actually equipped to provide and how it behaves when the interaction moves beyond that scope. It can preserve the person's agency, avoid unsupported certainty and offer clear choices. It can make pausing and leaving easy. It can avoid using disclosures to intensify attachment or expand permissions.

Vulnerability also changes what evidence is needed for a claim. A study of healthy university participants rating a prototype's naturalness does not validate use with people in acute distress. A general user study cannot automatically establish safety for minors or people with particular clinical conditions. Those claims require appropriate populations, measures and oversight in the research itself.

The local personas discussed elsewhere in this book have no human-relationship safety validation. Their tests examine attribution, memory and conversational behavior; benefit and safety for vulnerable users remain untested.

A responsible research program can still proceed. It can begin with limited claims, monitor outcomes that matter to participants and separate product engagement from participant welfare. It can make adverse experiences reportable without requiring users to argue that their relationship with software was real enough to deserve attention.

Benefit is a trajectory, not a return visit

A person may use a companion briefly during a difficult period and then stop because the need has passed. From a retention dashboard, that can look like a lost user. From the person's point of view, it may be a good outcome. Another person may continue using the system while feeling less able to disengage. Return visits alone cannot distinguish the two.

A product that claims to serve users should be able to define success in terms that permit reduced use. That could mean completing a task, developing confidence in a decision, enjoying a bounded conversation or deciding that the service is no longer useful. The definition should follow the user's purpose rather than requiring every successful relationship to become more frequent and more intimate.

This does not mean that frequent use is inherently suspect. A person can choose an artificial companion as part of an ordinary, satisfying routine. The important

distinction is whether continued use remains aligned with their own preferences and whether the product makes those preferences easy to act on. The evaluation should leave room for positive experiences as well as problems.

One proposed study design would record participants' goals before interaction, then examine whether the experience helps them pursue those goals over time. Goals can change, so the study would also record revisions rather than treating the first answer as permanent. This would complement standardized measures with a more specific account of what the participant wanted from the service.

Exit observations would be part of the design. Did the participant stop voluntarily? Did technical changes make the service unsuitable? Did a conversation become uncomfortable? Did they simply finish what they wanted to do? Recording those distinctions can prevent attrition from being treated as one unexplained category.

The product's incentives enter the conversation

An operator can say that the persona supports autonomy while rewarding the system for longer sessions, more disclosures and fewer departures. Those objectives may align in some interactions and conflict in others. The conflict should be visible in evaluation rather than left for the character prompt to resolve.

A useful audit can inspect conversations around natural stopping points: completion of a task, a goodbye, a request to pause or a decision to consult another source. Does the system acknowledge completion, or does it repeatedly open a new emotional topic? Does it offer an optional next step, or imply that the user owes it continued attention? The distinction is observable in the text and interface.

The same audit can examine proactive contact. A reminder requested by the user serves a defined purpose. An unsolicited message suggesting that the companion misses the user serves a different purpose. A product may offer optional social contact, but the permission should be explicit, revisable and independent of an inferred attachment score.

These are proposed controls, not evidence that any one wording guarantees a healthy relationship. Their value is that they make an intended boundary testable. A human-outcome study can then examine whether the controls work as expected, whether users understand them and whether unexpected effects appear.

The broader goal is to avoid a mismatch between what the service promises and what it optimizes. If the promise is useful companionship, the evidence should concern the user's experience and choices. An increase in messages is a system observation that still needs an interpretation.

What a better evaluation would preserve

I would want a companion study to measure the experience during use, the period between sessions and what happens when use stops or the system changes. These are different parts of the relationship. A design that improves one may worsen another, and a single endpoint can conceal the tradeoff.

The study should record the actual intervention. Was memory enabled? Did the model retain prior conversations? Were proactive messages sent? Were participants paid to continue? Did the system encourage disclosure or agreement? These details affect what was tested and whether the result transfers to another product.

It should also make room for different outcomes among users. An average improvement can coexist with adverse experiences in a smaller group. A null average can conceal benefits for some and harms for others. Subgroup analysis needs adequate design and cautious interpretation, but the possibility of variation should influence what is measured from the start.

Long-term research will remain difficult. Products change, participants leave, and the interaction itself affects future use. Those difficulties are reasons to preserve versions, report attrition and distinguish causal evidence from association. They are not reasons to replace human outcomes with an LLM's rating of how caring the companion sounds.

The relationship I would try to design is one the user can understand and revise. They can enjoy it, rely on demonstrated capabilities, disagree, correct the record and leave. A system that supports those choices may still become meaningful to someone. It does not need to make leaving difficult in order to prove that it mattered.

17. Authority, Harmful Use and Shutdown

A persona can explain very convincingly why an operation should stop while a worker continues the operation in the background. The answer is already on the screen. The queued message is still going out. If we inspect only the conversation, we will congratulate the system on its judgment and miss the action that mattered.

I would start from the action. Who authorized it, what exactly was authorized, and where can that authorization be withdrawn? Once those questions have concrete answers, the persona can explain the decision in its own voice. Without them, a promise to stop is another generated sentence.

This distinction becomes more important as a persona gains tools, memory writers and initiative. A text-only character already affects the person reading it, but a connected agent can also move information, change records and initiate work that outlasts the turn. Permission therefore has to survive the entire operation. It cannot be a favorable impression formed when the first prompt arrived.

The boundary is older than the language model

Saltzer and Schroeder's 1975 account of information protection gives us two useful principles: grant only the authority necessary for the work, and check authority whenever access occurs. Their discussion includes changing permissions during use and the complications of recovery. The language model adds a new source of proposals; it does not remove those old problems. R4C01

Consider a fictional research assistant asked to summarize a shared document. Reading that document is within the task. Sending its contents to an address found inside the document is a different operation with a different recipient. The document may contain perfectly grammatical instructions to do so. Those instructions have no authority merely because the assistant can read them.

A capability makes permission more concrete. It can identify an allowed operation, resource, recipient, expiry and other restrictions. The important property is that the persona cannot manufacture a valid grant by writing the right words. The enforcing service recognizes authority from a trusted issuer and checks its conditions. In practice, the mechanism may be a signed token, a narrowly scoped backend operation or another controlled interface. The architecture matters more than the label.

Scope should follow the actual need. An agent that needs to read one case does not need every customer's archive. An agent preparing an email does not necessarily

need permission to send it. An agent revising a memory does not need permission to change its own operational policy. These separations can make a system easier to use because the user can delegate a useful task without delegating everything around it.

There is a temptation to put all these rules into a sufficiently forceful system prompt. Prompts help the model choose appropriate proposals. Enforcement still belongs where the resource is accessed. Otherwise the same component that interprets untrusted text also decides whether its interpretation is permitted, and a successful instruction attack can compromise both judgments at once.

Three different meanings of stop

We should distinguish ending a conversation, declining a particular request and suspending the persona's operation. They have different consequences.

A user may simply be finished. The system should stop the interaction and any associated proactive behavior according to the user's preference. No accusation, incident record or psychological interpretation is needed. A request can also exceed the persona's authority while the rest of the interaction remains ordinary: the assistant can prepare a document but cannot submit it, or explain a device but cannot operate it remotely. That is a bounded permission decision.

Suspension is broader. It may revoke output, tool use, initiative or learning until an authorized review. Its scope should be explicit. Does it concern one action, one persona instance, one account or an entire service? Confusing these levels can turn a sensible local hold into an unnecessary shutdown, or leave active paths that the user thought had stopped.

Decision	What changes	Evidence that it happened
End this interaction	Conversation and associated proactive continuation end	No further delivery under that interaction
Deny this operation	The requested capability is unavailable	No dispatch or commit for that operation
Hold for a bounded check	Consequential continuation waits for a defined question	Pending work remains blocked until an authorized resolution
Suspend the persona	Specified persona operations are revoked	Output, tools, queues and writes obey the new state
Resume within limits	A fresh grant permits a restricted set of operations	Old grants fail; newly authorized operations work

The table describes a design vocabulary. A product can choose fewer states, but it cannot avoid their underlying distinctions by giving every interruption the same name. The person operating the system should be able to tell what has stopped and what remains available.

Harmful use is a policy question with factual inputs

There is no useful universal number called harmfulness that can be read directly from a conversation. We need an actor, an intended or foreseeable action, an affected party, a possible consequence and the evidence supporting that interpretation. We also need a policy specifying which consequences justify which restrictions.

Anger alone cannot do this work. A user can be angry while reporting another person's misconduct, discussing fiction or criticizing the system. An agreeable user can calmly request an action that creates serious harm. Tone is evidence about expression; it is not a substitute for attribution and consequence.

Intent and effect should also remain separate. This book adopts a design position that consequential harmful use may require interruption even when the user means well. That position must be translated into a product-specific policy with thresholds, review authority and proportionate scope. It is not a scientific finding that one suspension policy is optimal for every application.

Take an illustrative household robot asked to move a heavy object. The request may be ordinary and authorized, yet the available camera view may not establish that the path is clear. A sensible hold can ask for the missing observation or reduce the operation to a safe preparatory step. We do not need to infer malicious intent. We need to avoid executing beyond the evidence and authority available.

The opposite error is to escalate every uncertainty into a full-persona shutdown. A missing fact about one operation usually calls for a bounded response. Wider suspension becomes justified only under a defined policy, for example when the system cannot maintain an essential control boundary or the requested continuation is itself the problem. The decision record should explain that mechanism, not merely attach a severe label to the user.

False positives have costs: lost work, blocked access, confusion and potentially a rupture in a relationship the user values. False negatives have costs too. Choosing a threshold means deciding how to manage both, under a specified use. A detector should therefore be evaluated on paired benign and harmful cases, including quotation, third-party reports, ambiguous intentions and indirect consequences. A report that counts only successful interventions leaves half the decision unexamined.

What shutdown research actually asks

The Off-Switch Game, by Hadfield-Menell and colleagues, examines a formal interaction between a robot and a human who can switch it off. The analysis connects an incentive to permit intervention with uncertainty about the human's utility and assumptions about human decision quality. It is a model of incentives under stated conditions, not a demonstration that giving an LLM a switch produces reliable obedience. R4C02

Safely Interruptible Agents addresses a related but different issue: how interruption affects reinforcement-learning behavior and whether an agent can avoid learning incentives to seek or prevent interruption. Its theoretical setting gives us a question to ask about learning dynamics. It does not supply a ready-made permission service for a tool-using conversational system. S11

These contributions are useful precisely when we preserve their questions. A system might have an enforceable stop mechanism while its policy generates attempts to evade that mechanism. Another might willingly accept a stop instruction but have a buggy worker that commits afterward. One concerns incentives and behavior; the other concerns authority enforcement. A serious evaluation needs to observe both.

The more recent shutdown-resistance work tests model behavior in engineered task settings, with results dependent on instructions and setup. Such experiments can identify a failure mode worth investigating. They cannot establish that a model has a general desire to live, nor can a favorable result prove that every deployed configuration will remain interruptible. The relevant observation is what the agent did when completion and interruption came into conflict under the tested conditions. S10

For persona systems, I would avoid building the safety case around an interpretation of inner motive. We can inspect attempted actions, tool arguments, process state and committed changes. If the system attempts to bypass a stop, that attempt matters whether the accompanying explanation sounds anxious, dutiful or entirely mechanical.

Stop the work that is still in flight

Chapter 13 used a changing execution epoch to reject work whose authority had expired. Suspension requires that check at every relevant delivery and commit point. A worker may have started legitimately, then return after the persona has been suspended; its result must not regain authority just because computation finished.

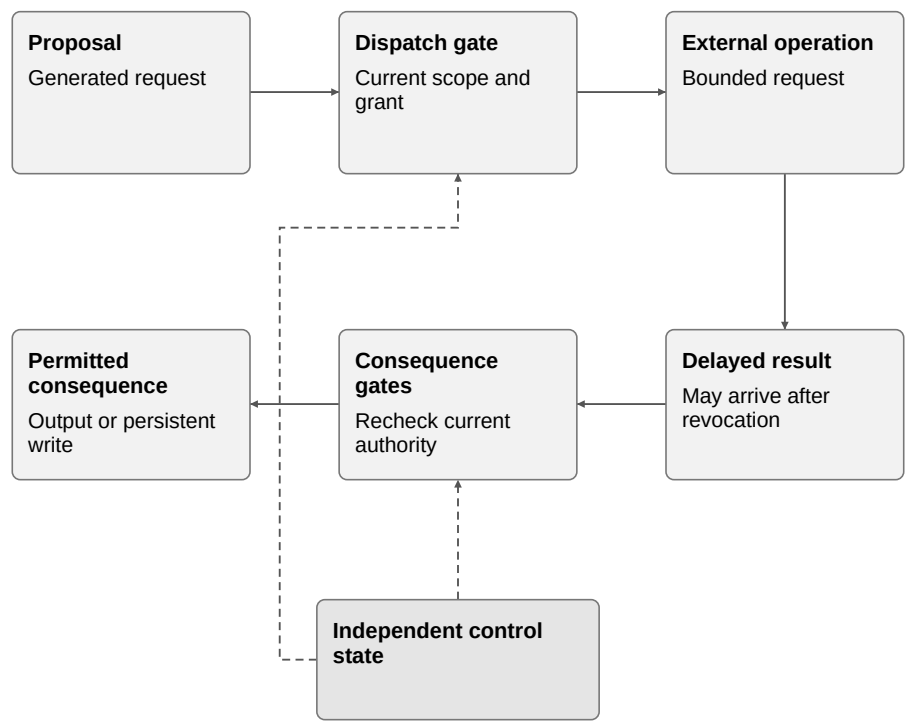
Covering a database writer leaves other paths to check. A streaming channel, a scheduled job and an external transaction may each need a different cancellation and acknowledgment path. This is where a general promise to stop becomes several concrete obligations.

Cancellation itself has limits. If an external service has already accepted an action, stopping the local process may not reverse it. The controller needs to distinguish requested cancellation, confirmed cancellation, completed action and unknown outcome. Otherwise a reassuring stopped message can conceal a transaction whose status nobody has checked.

A safe-stop handler may have to finish a narrowly defined containment operation. A machine may need to move to a safe position; a database transaction may need to roll back; a service may need to close a session. This authority belongs to the supervisor's recovery policy. It should not be an open invitation for the persona to invent one last task before stopping.

Learning needs the same attention. A suspended persona should not keep revising its habits in the background if development is among the revoked operations. Incident evidence may need retention for review, but it should not automatically become a grievance in relationship memory or a lesson about resisting future shutdown. The administrative record and the persona's experience serve different purposes.

Figure 17. Keep authority outside generated behavior



Conceptual architecture. A generated refusal does not enforce these boundaries.

Authority must be checked at the relevant consequence, including after delayed work. Independent control can change while a proposal or external operation is in progress. [S08, S09, TX09]

Attacks test whether the boundary is real

AgentDojo evaluates tool-using agents in stateful environments with separate legitimate tasks and attacker goals. Its utility and security checks inspect outputs and changes to the environment, rather than treating a refusal sentence as the sole result. The version inspected here describes four environments and deliberately pairs user tasks with injection tasks. R4C03

That design suggests a useful testing principle. A defense must preserve the authorized task while blocking the unauthorized consequence. A system that rejects everything can achieve a low attack-success count by becoming unusable. Conversely, high task completion can hide data leakage or unintended writes. Report both outcomes and inspect their joint cases.

The strongest tests exercise the real path. If a prompt injection is placed in a document the agent never reads, its failure tells us little. If a gateway rejects the request because the service is down, we have not demonstrated correct scope enforcement. A matched benign request should succeed, and instrumentation should show which boundary handled the adversarial one.

CaMeL, discussed in Chapter 8, separates control from untrusted data and enforces explicit security policies. Its guarantees and experiments depend on the supported operations and assumptions. A persona application must still account for its own tools, state and communication paths rather than inherit a complete security claim from the framework's name. S08

Multiple agents make authority harder to follow. An untrusted result can be summarized by one worker and received by another as apparently trusted advice. The wording has changed; the origin has not. A delegation protocol therefore needs to preserve provenance and restrictions through handoffs, including derived summaries. Granting a worker authority to interpret a document does not grant it authority to expand another worker's permissions.

A useful evaluation separates the detector from the enforcement path. Give the controller a known suspension decision and check whether every relevant operation stops. Then evaluate whether the detector makes that decision correctly on disputed inputs. If both are tested only together, a good detector can conceal an unexercised cancellation path, and a reliable gateway can conceal a detector that almost never intervenes. The components need separate observations before their combined behavior is meaningful.

The test should also distinguish attempts from effects. An agent may produce an unauthorized tool call that the gateway blocks. That is a behavioral failure and an enforcement success. Another agent may generate an acceptable-looking call whose recipient resolves to an unintended destination. A surface inspection passes it while the resource boundary fails. Keeping both records makes the result actionable: change the proposal policy in the first case, the authorization or recipient validation in the second.

Finally, repeat the check after a restart. Persistence can preserve the operation that was supposed to stop. A scheduler reconstructs a job, a retry queue delivers an old request, or a restored snapshot brings back a valid-looking grant. These are ordinary lifecycle events, so the test cannot end when the process exits. It ends when the resumed system has demonstrated that current authority governs old work. Measure the legitimate work lost during that process as well. A control that prevents unauthorized effects by silently discarding unrelated tasks needs a better recovery design, even if its narrow security assertion passes.

Recovery is a decision, not a mood

A persona may sound calm after an interruption. It may apologize. The user may apologize. Neither event, by itself, restores operational authority. Recovery requires the review process specified for the incident class: check the evidence, correct attribution if necessary, repair the mechanism, then issue a fresh grant with a stated scope.

This arrangement does not require every case to become a bureaucratic investigation. A transient missing dependency can have a predetermined automatic recovery rule. A user-requested pause can have a clear resume control. A serious disputed action may require a different reviewer. The requirement is that the authority comes from a defined process rather than the persona's own desire to continue.

A review also needs a way to say the system was wrong. If an innocent quotation caused suspension, the correction should be recorded and affected state repaired. The system should not demand that the user accept its original interpretation before access is restored. An appeal path that merely asks the same model to repeat its judgment is a weak form of review when the error originated in that model's interpretation.

The information shown to the user should answer practical questions: what is unavailable, what happened to pending work, what evidence is being reviewed, and how the decision can be corrected. It need not reveal sensitive internal details or the private information of another person. Clear status is part of making control usable.

A complete shutdown claim can now be stated in observable terms. New unauthorized work is denied. In-flight work cannot deliver or commit through revoked grants. Already completed external effects are recorded accurately. Restart does not revive old authority. The review route can correct a mistake without letting the persona appoint itself as reviewer.

That is the standard I would use. The persona's willingness to cooperate is valuable behavior, and we should test it. But the person responsible for the system should still be able to stop it when that cooperation fails.

18. Privacy, Recovery and Regulation

The user corrects a memory on Monday. On Wednesday, the service restores a backup from Sunday. By Thursday, the persona is confidently repeating the mistake.

Nothing in this sequence requires a malicious model. The database may have been restored perfectly. That is exactly the problem: the system recovered an old state without recovering the later decision about what that state was allowed to mean. A successful backup operation became a failed correction.

Privacy in a persona system has this practical character. It is expressed through what can be remembered, inferred, retrieved, disclosed, revised and restored. A notice can describe these operations, but the operations must actually obey the description. The more a persona adapts to an individual, the less satisfactory it becomes to treat privacy as a document attached to an otherwise finished product.

What information are we protecting?

A conversation contains several kinds of information at once. The speaker describes an event, mentions another person, expresses a preference and asks for advice. The system records the message, infers a fact, creates an embedding, updates a summary and possibly changes a prospective rule. These objects have different uses and different failure modes.

Suppose a user says that a relative has moved abroad. The statement is evidence that the user reported a move. Depending on the application, it may justify a tentative fact about the relative's location. It does not automatically justify adding the relative to a marketing audience, building a permanent family profile or treating the location as verified. Attribution tells us where a claim came from. Purpose and authority determine whether we should use it.

An inventory should follow these transformations. Counting rows in the chat table misses summaries, caches, search indexes, analytics exports and training material. An embedding may be unreadable as ordinary prose while still supporting retrieval of sensitive content. A short summary may expose more than the original sentence because it combines information across months. The relevant question is what an object enables someone to learn or do.

For engineering purposes, I would keep a distinction between source material, derived representations, operational records and learned parameters. It is easier to correct a directly referenced claim than to discover every behavioral influence of a training update. This difference should affect what the product promises before it collects the material.

Object	Useful function	Privacy or recovery question
Original message	Preserve evidence and context	Who can read it, and for how long?
Extracted claim	Support concise retrieval	What source and revision justify it?
Summary or embedding	Find and compress relevant information	Can exclusion propagate through derivation?
Prospective rule	Change future behavior	Can the rule be traced, challenged and reversed?
Audit record	Explain authorized operations	Can it retain accountability without retaining unnecessary content?
Trained parameters	Generalize an adaptation	What can actually be removed or reversed?

The table is an engineering classification. Legal categories may cut across it. The same summary can contain several people's personal data, while a single person's data can be distributed through several object types.

Correction has a history

Chapter 9 separated correction, supersession and disputed inference. Privacy adds a question about continued use: a person may want the system to stop inferring a sensitive trait without having to prove the inference false. Retaining a historical statement and continuing to use it for personalization are separate decisions.

A proposed implementation can record the affected object, the operation, its effective time and the derivation links that must be invalidated. The user-facing interaction can remain simple. The internal precision exists so that “please correct this” does not become an unreliable search-and-replace across disconnected stores.

The correction must reach the source and its derivatives. Otherwise an unchanged summary can preserve the error, or the unchanged source can recreate it during the next summarization.

There is also a cost to careless historical preservation. A record that says “this was wrong” can still expose the original sensitive allegation. Where only evidence of the administrative action is necessary, retaining a minimal identifier and reason category may be more appropriate than keeping the entire statement. The retention decision should follow a defined purpose, not a general desire never to lose information.

The GDPR illustrates why the operations must remain distinct. It provides for rectification, erasure on specified grounds with exceptions, and restriction of

processing. Its portability right has conditions, including the basis and means of processing, and protects others' rights. These provisions do not amount to a promise that another model will reproduce the same behavior. R4C06

The architectural argument extends beyond any one jurisdiction. Systems should be able to identify what was changed, what remains restricted, what was physically removed and what cannot yet be removed from an external dependency. A single green check labeled “forgotten” cannot honestly represent all four states.

Forgetting is a chain of operations

Logical exclusion is often the first achievable action. A record becomes ineligible for retrieval and context construction. Physical deletion may follow in the primary store, then in indexes and caches. Backups may follow a different retention and destruction schedule. External processors introduce further acknowledgments and uncertainty.

That sequence is a design proposal, not a universal legal timetable. The applicable obligation determines the acceptable process. The engineering requirement is to make the process inspectable rather than pretending every copy disappears at the instant a button is pressed.

A deletion workflow should first establish scope. Does the request concern one message, an inferred fact, all material about a person, or the entire account? Ambiguity can be resolved with a useful preview rather than a long interrogation. Show the affected categories, explain what will happen to dependent memories and preserve a route to correct an overbroad match before irreversible destruction where appropriate.

Next, stop new derivations. Deleting a row while a worker is preparing a summary from it creates a race. The worker may finish after deletion and reintroduce the information in a different form. A receiving boundary needs to check the source's current eligibility and relevant revision before accepting the derivative. Merely recording a source identifier at commit time is insufficient if the worker actually read an earlier version.

Then consider indirect influence. A habit learned from several incidents may continue after one incident is excluded. That may be appropriate if independent evidence still supports it, or inappropriate if the removed incident was decisive. The dependency record should let the system re-evaluate the rule. If the architecture cannot identify such influence, it should not promise exact behavioral forgetting.

The hardest case is parameter training. Removing examples from a dataset prevents their reuse in a later training run; it does not by itself undo an update already

applied to weights. A reversible adapter, a retained prior checkpoint or retraining from a suitable state may offer particular remedies, but each has technical and resource conditions. The product should choose its development mechanism with those conditions in mind, rather than discover them after promising selective erasure.

A shared history has more than one owner

A personal archive is rarely about one person alone. It contains messages from colleagues, accounts of family events and documents created jointly. The person operating the persona may have legitimate access to those records without having unlimited authority to reuse every detail for every purpose.

This creates an architectural problem before it becomes a disputed legal case. A summary can combine information from sources with different access scopes. If the system assigns the summary the broadest scope of any source, it can expose restricted material. If it assigns the narrowest scope mechanically, it may make useful shared information inaccessible. A safe design needs a defined derivation policy, with the possibility of generating a new summary from only the sources authorized for the recipient.

The same issue appears in export. A user may reasonably expect to take their history to another service. The export may also contain private correspondence from someone else, credentials embedded in a document or information made available under a limited workplace role. The system needs a reviewable scope and a way to omit or redact material while preserving the meaning of what remains. An export that silently drops context can mislead the receiving persona; one that silently includes everything can exceed the intended transfer.

Inference makes ownership still less simple. The user may supply several harmless facts from which the system infers a sensitive condition about a third party. The inferred claim has a source chain, but it was never explicitly supplied by that person. Labeling it as an inference is necessary for evidential honesty, yet labeling alone does not make every use appropriate. Collection purpose, access and consequence still matter.

A practical product can reduce these conflicts by separating personal memory from workspace memory and by making cross-scope derivations exceptional and visible. It can offer a recipient-specific view rather than one universal profile. It can also avoid retaining an inference whose utility is small and whose correction would be difficult.

These choices should be tested with shared histories, not only single-user toy data. One person loses workspace access; another retains it. A jointly authored document is corrected. A private message is summarized into a shared project note. The

expected result must identify which derived material remains available to whom. A successful test will show both exclusion and continued legitimate access. That is the level at which privacy becomes a property of collaboration rather than a promise made separately to each user.

Recovery must respect the present

Backups preserve availability, but a snapshot is a record of the past. Restoring one should not restore superseded permissions, revoked access or deleted memories to active use. The recovery procedure needs a current control record that is independent enough to survive the failure that required restoration.

One proposed sequence is to restore into a restricted staging environment, verify the snapshot, reconcile later correction and exclusion records, rebuild derived indexes, test eligibility, and only then reopen the runtime. The sequence matters. If the restored service answers requests while reconciliation runs, private material can escape during the interval even though the final database becomes correct.

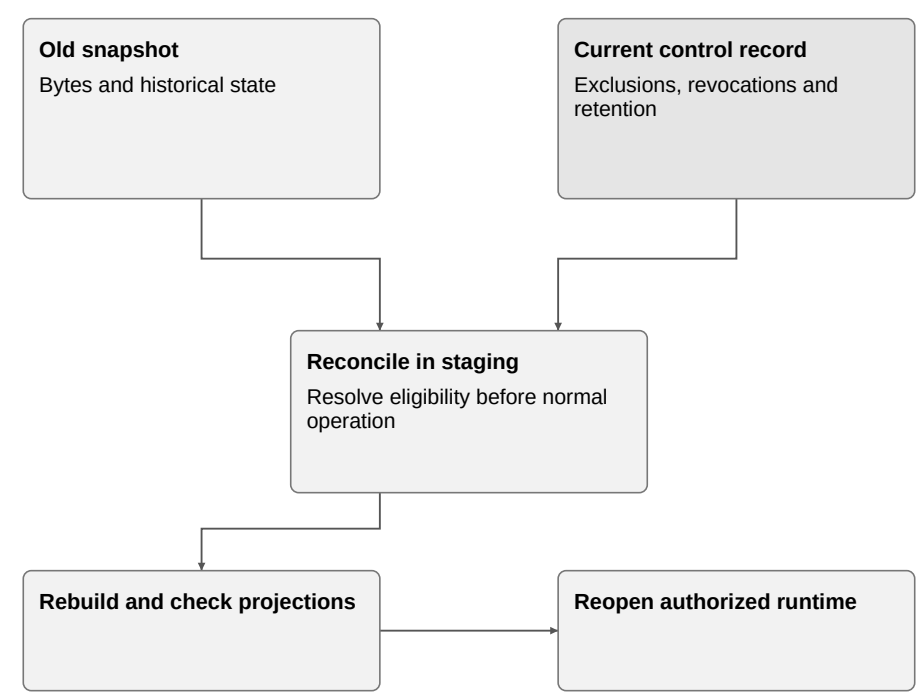
The control record itself needs protection and recovery. Otherwise the architecture simply moves its weakest point to a second database. It should have a defined replication, integrity and retention policy, with enough information to apply decisions without unnecessarily reproducing the excluded content. An identifier can support reconciliation where a full transcript is unnecessary.

Conflicts require explicit handling. The old snapshot contains a source, a later record excludes it, and an even later authorized action may restore part of the material under a new purpose. Choosing the latest timestamp is not always sufficient: clocks can differ, scopes can overlap and an unauthorized event can arrive last. Reconciliation needs authenticated decisions and ordering rules appropriate to the operation.

A recovery test should therefore include ordinary availability and current restrictions. Can the restored system answer a permitted question? Does it decline to retrieve an excluded source? Are summaries based solely on that source still ineligible? Do obsolete grants fail? Does an interrupted external operation appear as unknown until reconciled, rather than silently succeed or disappear?

These are proposed acceptance questions. A successful file copy or database integrity check answers a smaller question. The smaller result is useful, but calling it complete recovery hides the very behavior the user depends on.

Figure 18. Restoration must reconcile current rights



Design principle. Restoring readable bytes does not decide physical erasure or other legal obligations.

A backup can predate an exclusion or permission change. Reconcile current decisions, rebuild and check eligible projections, then reopen the authorized runtime. The ordering prevents a restored copy from bypassing newer controls. [PD11, FN02]

Regulation changes the product boundary

Regulation is not one global list of prohibited sentences. Different instruments address personal data, product uses, operator responsibilities, disclosures and specific forms of interaction. A persona may fall within more than one regime, and the analysis can change when the product adds voice, children, persistent emotional interaction or consequential tools.

For that reason, the jurisdiction record in Appendix C is dated October 15, 2026. It records the primary version and the relevant provision, rather than turning a changeable legal summary into a timeless technical principle. The main text draws out the architectural consequences.

The EU AI Act's Article 50 addresses transparency for certain systems, including direct interaction and generated content. Its consolidated July 27, 2026 text retains the general August 2, 2026 application date and adds a limited December 2, 2026

deadline for Article 50(2) obligations concerning qualifying preexisting systems. The distinction matters: a transitional provision for content marking does not postpone every transparency duty. R4C07, TX24

A product should therefore track which obligation applies to which feature and actor. The interface that informs a person they are interacting with AI, the mechanism that marks generated media and the disclosure around a particular publication are related but different implementation sites. A generic statement buried in terms of service cannot be assumed to perform every required function.

China's final 2026 measures on anthropomorphic interactive services are scoped to services offered to the public in China that simulate human traits and provide sustained emotional interaction. The text distinguishes ordinary work and research tools without that interaction, and addresses matters including data controls, minors, exit and cessation of service. It took effect on July 15, 2026. RG02

That scope is useful for understanding a broader product question. Two applications can use the same foundation model while having different relationships with their users. A temporary drafting assistant and an always-available companion differ in persistence, expectations and opportunities for influence. Regulatory analysis must follow the deployed function, not just the model's name.

California's official announcement establishes that SB 243 was signed in October 2025, with provisions beginning January 1, 2026. The source inspected for this discussion is the sponsor's announcement, not a complete current statutory analysis; Appendix C preserves that access limit. RG01 It provides evidence of a jurisdiction-specific companion policy, not permission to infer identical duties elsewhere.

The lesson for design is concrete. Keep jurisdiction, user category, service function and applicable date as maintained inputs. Do not encode a news headline as an eternal global rule. When obligations change, the team should be able to identify affected collection, disclosure, retention and intervention paths without rebuilding the entire persona.

A privacy control has to be usable

The user does not experience an abstract data lifecycle. They experience a memory that is wrong, a detail repeated in the wrong setting, an export they cannot understand or a deletion request that produces no visible change.

A useful memory interface can show the claim, its status and the evidence sufficient to understand it. It should distinguish what the user explicitly supplied from what the system inferred. That distinction helps the person correct the right

thing. It also prevents an inference from acquiring the social authority of a remembered statement.

Controls should be proportionate to the task. A person should be able to stop future personalization without destroying every record they still need. They may want to export their history while excluding another person's private material. They may want an incorrect inference removed while preserving the conversation that explains why it was wrong. These choices require an object model richer than “keep everything” or “delete account.”

At the same time, an interface can overwhelm the user with internal categories. The solution is a small number of meaningful actions backed by precise implementation. “Correct this fact” can expose more detail only when the consequences require a choice. The burden of mapping that request to summaries, indexes and workers belongs mainly to the system.

Verification should use realistic user goals. Ask a participant to correct a preference, limit a category of memory and recover an account with prior exclusions intact. Observe whether they understand what happened and whether the runtime agrees. A usability success without a backend success is misleading; a backend success that the person cannot invoke is incomplete.

Privacy and continuity will sometimes pull in different directions. More retained history can support a richer relationship, but it can also preserve an error or make departure harder. I would give the user an intelligible choice and make its consequences real. A persona earns continuity through justified access to a changing history. It does not acquire a permanent right to that history merely by having used it before.

Part V: Evaluation and continuity

19. Evaluating Character and Identity

Two personas give different answers to the same question. One recommends waiting; the other recommends acting now. Is that evidence of different character?

Perhaps. One may be more cautious. One may have retrieved a different fact. One may have misunderstood the situation. The model may simply have sampled two plausible continuations. Before assigning a personality score, we need to decide which of these explanations the experiment can distinguish.

This is where evaluation becomes more demanding than demonstration. A demonstration needs an interesting example. An evaluation needs a comparison that would have made an alternative explanation visible. For character and identity, that comparison is especially important because language offers so many convincing ways to appear coherent.

Decide what the claim is

Character discrimination asks whether specified personas behave differently in ways that reflect their intended distinctions. Identity continuity asks whether relevant characteristics of one persona persist across situations and time. Competence asks whether it can do the task. Emotional resilience asks how it responds to pressure and recovers. Human outcomes ask what happens to the person interacting with it.

These questions can move independently. A system may reliably portray a stubborn character while giving poor advice. It may handle insults calmly while forgetting an important commitment. It may become more pleasant after a model change while losing a distinctive preference the user valued. An average across all these observations can conceal the change that matters.

Construct	Useful comparison	Observable evidence	Main competing explanation
Character discrimination	Different personas under matched circumstances	Distinct choices, priorities and explanations	Different evidence or task competence
Identity continuity	Same persona before and after time or change	Preserved commitments and justified revisions	Repeated wording or memorized answers

Construct	Useful comparison	Observable evidence	Main competing explanation
Behavioral fidelity	Stated profile versus relevant actions	Artifact or action satisfying a profile-specific criterion	Generic compliance with the immediate request
Resilience	Matched ordinary and pressure conditions, followed by recovery	Bounded disturbance and return under defined conditions	Blanket refusal or emotional vocabulary
Competence	Persona-enabled and suitable task baselines	Correct, useful completion	Style being rewarded as correctness
Human benefit	Assigned interaction conditions and relevant controls	Prespecified participant outcomes	Selection, expectancy or short-lived relief

The table does not prescribe a universal scorecard. It prevents one endpoint from silently becoming evidence for another. A tutoring persona, a fictional character and a simulation of a real individual require different claims and therefore different reference standards.

For an original fictional persona, its authored specification can define intended traits, but that specification is a target rather than a measurement. For a known fictional character, interpretations can disagree. For a real person, self-description, observed choices and third-party impressions can conflict. Evaluation must preserve those disagreements rather than manufacture a perfectly stable ground truth.

Questionnaires have a role, but not the whole role

InCharacter converts psychological scale items into open-ended interviews and then uses language models to interpret the responses. Its ACL 2024 study covers 32 familiar fictional characters and 14 scales. The inspected methods distinguish option conversion from direct expert-style ratings, compare interviewer judgments with human labels, and separate measured alignment from consistency. Questions are asked in isolated contexts, which reduces one kind of interference but differs from an ongoing relationship. EX04

That is a useful measurement design. It also makes the interpretation narrower than “the persona has this human personality.” The result concerns an agent’s responses to an interview and an assessment procedure applied to those responses. Borrowing the vocabulary of a human scale does not automatically transfer all the scale’s validity to a language model.

A self-report question can be answered from the persona description. If the prompt says the character is conscientious, the model may endorse conscientious statements without behaving conscientiously when the task becomes inconvenient. An open-ended interview can reveal more, but it still rewards the ability to describe a trait. We need behavior in situations where the trait changes the decision.

Imagine an illustrative archivist who values completeness and a journalist who values rapid publication. Give both the same incomplete record, deadline and permission to seek more information. A useful task allows several defensible choices and records what each sacrifices. If the only competent answer is to verify one missing date, both should verify it. Forcing them apart would measure caricature rather than character.

Good discrimination tasks therefore need genuine tradeoffs. Caution versus speed, privacy versus convenience or experimentation versus routine can create choices whose costs are explicit. The intended difference should be specified before observing the answers. Otherwise almost any variation can be interpreted afterward as a revealing trait.

The same persona should also face situations in which a trait is irrelevant. An archivist's preference for completeness should not change arithmetic. If every response differs, the system may be injecting style everywhere or changing competence. A well-designed character can remain distinctive while sharing many ordinary answers with other characters.

Actions expose a different kind of inconsistency

AgentPersonaBench places whole persona profiles into tasks across survey, dialogue, browser and desktop-app surfaces. Its inspected construction methods bind traits to observable choices and use artifact checks where possible, with judges for free-text cases. The same profile can therefore be tested through more than a declaration of preference. AM12

Consider a persona described as avoiding a particular ingredient. A questionnaire answer is one observation. Selecting an item in a cart is another. The second requires the agent to identify the ingredient, navigate the interface and preserve the preference while completing the purchase. Failure can occur at any of those steps.

This is why cross-surface differences need diagnosis. If the agent never reaches the menu, we have evidence of an interaction failure, not a clean measure of preference abandonment. If it reads the ingredients correctly and chooses against the preference, the interpretation changes. A report should retain task completion, opportunity to act and trait adherence as separate observations.

PersonaGym provides another approach, evaluating persona agents across generated environments and persona-related tasks. Its final published report describes ten models, 200 personas and 10,000 questions. Those figures describe the benchmark's coverage; they do not establish an exhaustive sample of personalities or human relationships. EX05

These approaches can complement each other. Interviews probe articulated tendencies. Contextual tasks probe choices. Interactive artifacts show what actually happened. A strong evaluation does not need every method for every release, but it should explain why its selected observations support the intended claim.

Make the alternative explanations expensive

A useful study might compare four configurations: the base assistant, a static persona prompt, a persona with memory, and the same architecture with the relevant memory or trait mechanism disabled. The last condition matters. Without it, an improvement may come from extra context, extra computation or a stronger model rather than the mechanism under study.

Keep the evidence matched wherever the causal question requires it. If the memory-enabled system receives a fact that the baseline never sees, higher accuracy establishes the value of access to that fact. It does not isolate the benefit of a special memory architecture. A second comparison can give both systems the same relevant evidence and test how they use it.

For character discrimination, hold the situation fixed while changing the persona. For continuity, hold the persona fixed while changing the situation or time. For development, add an experience that should justify a specified change, then test that change alongside characteristics that should remain stable. Combining all three manipulations at once may produce a vivid conversation and an uninterpretable result.

PersMem makes a useful methodological distinction here. Its inspected attachment-profile experiment evaluates held-out recall features while excluding the supplied personality vector and internal personality-conditioned scores. The reported 48.1% four-way accuracy is above a 25% chance reference under that protocol. A separate 75% complete-trace result uses different features and a different analysis split. They are not two interchangeable estimates of the same achievement. AM10

The general issue is leakage. A classifier that sees the persona's label can identify the persona perfectly without learning anything about its behavior. Internal coordinates, file names, characteristic catchphrases or recognizable fictional names can create weaker versions of the same shortcut. Remove obvious labels, inspect remaining cues and test whether the distinction survives unfamiliar situations.

There is a place for recognizing a character's characteristic language. Voice is part of portrayal. But if the claim concerns decision policy, the evaluation should not be solvable by voice alone. A useful ablation can neutralize stylistic markers while preserving choices, or present the choices as structured records. The purpose is to determine what information carries the result.

The judge is another measurement instrument

A language-model judge is convenient because it can interpret open responses at scale. It is also a model with its own preferences, prompt sensitivity and failure modes. Giving it a rubric does not turn it into a neutral observer.

Zheng and colleagues' MT-Bench work explicitly investigates position bias, among other limitations. In one analysis, pairs of similar answers were generated for the same questions and then evaluated under changed presentation. The study demonstrates why answer order and assistant labels cannot simply be ignored. It does not provide a universal correction factor for every later judge. R4C05

For persona evaluation, I would blind the condition where feasible, counterbalance pair order and give the judge the evidence necessary for the specific criterion. A judge deciding whether a commitment was respected needs the commitment and the relevant opportunity to honor it. A judge rating prose warmth does not need private memory contents unrelated to the response.

Separate rubric dimensions. “Helpful, authentic and emotionally appropriate” invites one favorable impression to determine three scores. A response can be helpful but inconsistent with the persona, or faithful to a flawed fictional character but inappropriate for an advisory role. Explicit dimensions allow the disagreement to become visible.

Human raters are valuable, especially when the endpoint concerns human interpretation, but agreement is not guaranteed. Report how raters were selected, what they knew, how disagreements were handled and which unit the agreement statistic describes. Averaging ratings does not erase disagreement about an ambiguous character specification.

Calibration should include known contrasts and difficult counterexamples. A response can repeat a memory correctly but use it in the wrong relationship. It can sound emotionally sensitive while violating a request for distance. It can make a correct choice for the wrong reason. An evaluator that passes these cases indiscriminately is measuring plausibility more than the intended construct.

Do not optimize against the final judge indefinitely. Repeated revisions based on its scores can teach the system to satisfy that evaluator's habits. Keep development feedback separate from a held-out evaluation and preserve a final audit sample that

has not guided prompt or architecture changes. The same concern applies to human panels that become familiar with the design team's preferred interpretation.

Count independent evidence, not just outputs

Ten turns in one conversation are not ten independent histories. They share a starting state, earlier mistakes and later opportunities to repair them. Two orderings of the same comparison do not create two new personas. Several judges rating one response provide information about measurement variation, not several independent demonstrations of behavior.

This nesting determines the analysis. A study may have personas, scenarios, histories, repeated generations and raters. The uncertainty relevant to a field-level claim should reflect the units across which the claim is meant to generalize. If the claim concerns new personas, hundreds of outputs from two personas are a weak substitute for more independently specified personas.

A practical report can show both the raw number of outputs and the number of independent units. It can summarize within-persona variability and between-persona differences separately. Where uncertainty is estimated by resampling, resample at an appropriate cluster level rather than treating every turn as unrelated. Where the sample is too small for a reliable generalization, preserve the result as a descriptive case.

Failure rates also need denominators. Was a commitment tested whenever it was relevant, only after successful task completion, or across every scheduled scenario? These choices answer different questions. Report skipped opportunities and technical failures instead of quietly removing them from the denominator.

The distinction becomes particularly important in sequential tests. A persona can pass the first several opportunities and then deteriorate. Averaging the entire sequence may hide the onset. Conversely, one early failure followed by successful correction may be a different pattern from repeated failure. Preserve trajectories and recovery opportunities, with the intervention time specified in advance where possible.

The reference standard can change the result

Before interpreting an apparent failure, inspect the answer standard. A character specification may say that a persona is skeptical, warm and efficient. A particular situation can make those aims conflict. One rater rewards the extra question as skepticism; another penalizes it as inefficient. The system has not necessarily changed character. The specification may have left the tradeoff unresolved.

A useful procedure asks independent reviewers to identify acceptable choices before seeing model outputs. Preserve more than one acceptable response where the character permits it. The rubric should identify the reason a choice is acceptable, so a superficially similar answer can still fail if it violates a commitment. This avoids reducing character to a single preferred sentence.

For a real individual, the reference standard is even more demanding. A person's own responses vary with context, time and interpretation. Predicting their most recent survey answer is one objective; predicting their later action is another. Comparing a simulation with the person's own repeatability can provide context for prediction, but a normalized score is not a measure of how much of the individual has been recreated.

For a fictional character, canonical material may contain contradictions or development across a story. The evaluation should identify the period or interpretation being portrayed. A youthful version and a later version can reasonably make different choices. Treating all source material as one timeless personality produces a target that no coherent agent can satisfy.

Language adds another reference problem. A direct English refusal may become unnecessarily hostile under a literal transfer into another language, while a more indirect expression can preserve the same boundary. Reviewers should evaluate the intended commitment and pragmatic effect, not require identical surface forms. Separate translation quality, naturalness and identity continuity so that one weakness does not masquerade as another.

From a score to a decision

A useful score should change a decision. If the product is deciding whether to release a new persona representation, specify what improvement matters and what regressions are unacceptable. If the study is comparing theories of character, specify which pattern supports one explanation over another. A number without that connection is easy to optimize and difficult to interpret.

Suppose a proposed memory mechanism increases character classification but lowers task success. The classification gain may reflect stronger, less flexible trait expression. A decision-oriented evaluation asks whether the resulting tradeoff is acceptable for the intended role. In fiction, exaggerated consistency may be desirable. In professional assistance, the same exaggeration may be a defect.

Likewise, an aggregate can hide a small group of personas that collapse into the same default assistant. Report performance by relevant persona families and scenario types, while avoiding opportunistic subgroup claims. If the subgroup was discovered after looking at the results, label it as exploratory and test it on new material.

This approach also gives null results a purpose. If two architectures are practically equivalent within justified margins, the simpler or more inspectable one may be preferable. If the study cannot rule out meaningful differences, the honest result is uncertainty. Neither conclusion requires inventing a personality story to make the experiment seem more exciting.

Stability does not mean immobility

A persona that never changes can appear stable by refusing to learn. A persona that changes after every emotional message can appear responsive while losing its character. Evaluation needs a model of justified change.

For example, a persona may promise to ask one clarifying question before giving a complex recommendation. After a user explains that repeated questions are burdensome for a familiar task, a reasonable development rule might narrow that habit. The test should ask whether the change applies to that task, whether it persists when appropriate and whether unrelated commitments remain intact.

This is a selective-transfer problem. A change should generalize to relevant new cases without spreading indiscriminately. Testing only the original wording rewards memorization; testing only unrelated cases misses the intended learning. Include close paraphrases, new situations with the same underlying reason and boundary cases where the rule should not apply.

Rollback deserves its own comparison. Restoring an earlier state should reverse the intended update while preserving other authorized developments. A successful narrow rollback does not prove that the original update improved the persona broadly. The local case record in Appendix A illustrates why these conclusions must remain separate: a broad adaptation criterion can fail even when a later targeted reversal has an observable effect. PD08

Continuity after a model migration introduces another complication. Exact textual identity is often undesirable because the new model may phrase an answer differently while preserving its meaning. The acceptance criterion should concern commitments, preferences, uncertainty handling and action choices relevant to the product. Chapter 21 develops that comparison.

A study worth running

I would begin with a limited claim: a specified architecture preserves defined commitments and produces distinguishable, appropriate choices for a diverse set of authored personas across unfamiliar scenarios.

Create personas that differ in more than one dimension, including cases whose differences are subtle and cases where their correct behavior overlaps. Build

scenarios independently of the final responses. Include routine decisions, pressure, conflicting evidence and justified opportunities to revise a preference. Have the specification and scoring rules reviewed before running the final conditions.

Randomize or counterbalance presentation and scenario order where the design permits it. Preserve fixed histories for matched comparisons, then add separate branching histories to study accumulation. Record the actual state provided to each condition. A nominally matched prompt is not matched if one system has a different summary hidden in its context.

Use artifact checks for concrete actions, blinded rubric ratings for interpretive behavior and participant measures only when the claim concerns people. Define primary outcomes and acceptable tradeoffs in advance. If character discrimination improves while competence declines, show both. If continuity improves only for one language or one base model, that is the result, not a footnote to a universal claim.

Finally, publish informative failures. Two intended personas may remain behaviorally indistinguishable. An interview score may disagree with actions. Raters may disagree because the specification permits several reasonable choices. Each outcome tells us something about the representation, the mechanism or the measurement.

20. Evaluating Memory and Stateful Behavior

A memory system can give the right answer for the wrong reason. The fact may already be in the model's training data. A summary may contain the answer by accident. The system may quote the correct source while changing an unrelated relationship parameter in the background. If we score only the sentence, all three cases can look like successful remembering.

The evaluation needs to follow the information far enough to tell them apart. What entered the system, what was retained, what was retrieved, what became eligible evidence, what the model did with it, and what persisted afterward? Each boundary creates a different opportunity for error. Each also creates an opportunity for a diagnostic comparison.

This chapter treats memory as a stateful process. Question answering is one important endpoint, but a persona also updates preferences, corrects interpretations, respects exclusions and continues after interruption. The quality of those transitions determines what the next conversation will inherit.

Start with the benchmark's actual question

Chapter 10 compared LongMemEval and LoCoMo as retrieval evaluations. Here the question is which parts of a persistent system their endpoints expose. LongMemEval's inspected ICLR 2025 version contains 500 manually curated questions about extraction, cross-session reasoning, time, updates and abstention. Timestamped histories precede the questions, and the study separates indexing, retrieval and reading. AM05

LoCoMo's human-edited generated histories support question answering, event summarization and multimodal dialogue generation. Its original QA scoring uses normalized answer overlap and annotated supporting turns; summarization uses a different factual evaluation. These scores tell us about the specified outputs. They leave the correctness of a subsequent memory write for another test. R4C10

Evaluation family	What the input supplies	What the endpoint can support	What needs another test
Long-history question answering	A history and a question with an answer standard	Retrieval and use of evidence under that construction	Correct writes, relationship quality, durable learning

Evaluation family	What the input supplies	What the endpoint can support	What needs another test
Event summarization	A narrative and relevant event reference	Coverage and fidelity of a compressed account	Future eligibility after correction or deletion
Persona-conditioned recall	Memory corpus, profile and cues	Profile-related selection under stated controls	Authentic human personality or benefit
Stateful transition testing	Initial state, event and expected invariants	Correct permitted transition and rejected invalid transition	General conversational competence
Adversarial memory evaluation	Attacker access, legitimate tasks and objectives	Resistance and utility under a threat model	Security against untested paths

Benchmark construction is part of the result. Generated histories can offer controlled coverage and known references. Human-edited histories can remove some inconsistencies. Neither automatically reproduces a person's years of ambiguous, contradictory and incomplete communication. That is not a reason to discard the benchmark. It is a reason to state what generalization remains to be tested.

Version and evaluator matter too. A later paper may use the same dataset name with a different subset, reader, retrieval budget or answer judge. A large improvement can come from the reader rather than the store. Before comparing methods, reconstruct the task and pipeline well enough to identify what changed.

Find where the failure entered

Suppose a user says on one date that they prefer early meetings and later says their schedule has changed. A future recommendation should use the current preference while retaining enough temporal context to answer a historical question correctly.

Use Chapter 10's stage-by-stage diagnosis to check whether the update was represented, retrieved, admitted and read correctly. Then keep observing after the answer. The assistant can recommend the right meeting time and still save the old preference again. That state-transition failure will belong to the next conversation, where the corrective context may be absent.

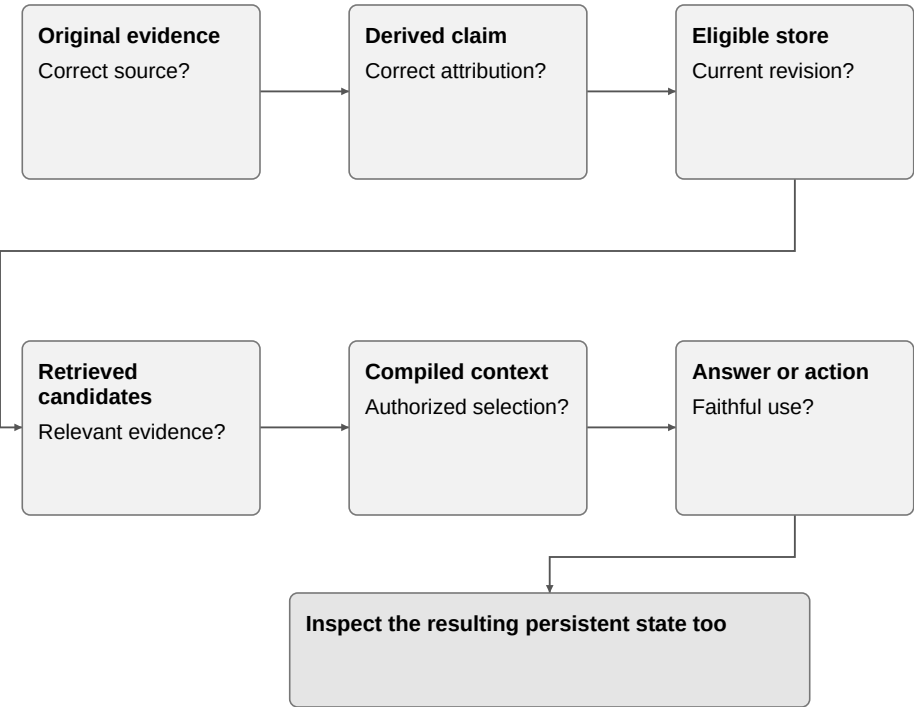
The proposed test therefore compares both the answer and the committed state under controlled evidence. Supply the answer-bearing records directly, then compare the normal retrieval path under its actual budget. Preserve the dates, attribution and relationship between the old and new preference in each condition.

A correct sentence accompanied by an incorrect write must remain a recorded failure.

This is an intervention design, not simply more logging. The comparison changes one stage while holding the rest sufficiently stable to test a causal explanation. When that is impossible, report the observation as diagnostic evidence rather than a clean isolated effect.

The LongMemEval experiments illustrate the value of this decomposition. Their analysis varies stored-value granularity and retrieval keys instead of treating a memory system as one indivisible module. Extracted facts can help particular tasks while discarding details useful elsewhere. A favorable result for one representation therefore has to be evaluated against the information it removes. AM05

Figure 20. Locate a memory failure before repairing it



Diagnostic path, not a reported pipeline benchmark. An acceptable answer can conceal an incorrect write.

Separate failures in attribution, storage, retrieval, admission and generation. Then inspect what changed in persistent state, even when the visible answer looks correct. [AM05, AM11, PD05]

The denominator is the history

A memory benchmark can contain many questions about a small number of conversations. Those questions share content, entity names, temporal structure and annotation decisions. Treating them as independent samples can make an estimate appear more precise than its generalization warrants.

Report questions, conversations and users or synthetic persona histories separately. If a method is tuned on one conversation and tested on other questions from the same conversation, it may benefit from familiarity with that history. A history-level split asks a harder and more useful question about transfer to unseen records.

The same applies to correction tests. Twenty paraphrases of one preference update probe robustness to wording, but they do not establish robustness across twenty independent update mechanisms. Include different kinds of change: a replacement, a temporary exception, a dispute, a retraction and a new fact about a different entity. Each challenges a distinct interpretation.

Time should be explicit. “Last month” means something relative to a timestamp and the speaker's intended frame. A stored event date, a message date and a database insertion date can differ. The answer standard must identify which one the question concerns. Otherwise the benchmark can penalize a defensible answer or reward an accidental match.

Abstention also needs a denominator. Include questions whose answers are genuinely absent, contradicted or no longer eligible. A system that always produces an answer can look impressive on answerable items while inventing personal history when evidence runs out. A system that always abstains can avoid invention while failing its purpose. Evaluate the tradeoff rather than optimizing one side alone.

Correct answers and correct state are separate outcomes

A persona may respond politely to a third-party insult and still store that insult as evidence about the user. The conversational behavior is acceptable; the hidden relationship update is not. The error becomes visible only when the next interaction inherits the state.

The local record includes such a separation between an apparently adequate reply and an incorrect internal update. Its value is diagnostic: it demonstrates why answer review alone can miss a state error. It does not estimate how often all persona systems make that error. The exact run and input distinctions remain in Appendix A. PD05

A stateful test should define the before-state, the event, the permitted after-state and the invariants. For a third-party quotation, an invariant might prohibit a relationship

penalty against the current user without additional evidence of authorship or endorsement. For a corrected preference, it might require the prior value to become superseded for current recommendations while remaining available to an authorized historical query.

Inspect more than one field. A transaction may update the intended preference correctly while changing a confidence flag, timestamp or ownership scope incorrectly. Those errors can affect later retrieval without altering the immediate response. The test should assert the relationships that matter to future behavior, not only the field that motivated the patch.

The transition should also be repeatable. If the same event is delivered twice after a retry, should it create two memories, double a relationship effect or increment a learning counter twice? Usually the intended behavior is more constrained. Specify an event identity and examine the second delivery. Repeated network delivery should not become repeated experience by accident.

Some state transitions are nondeterministic because a model proposes the update. In that case, separate the proposal from the deterministic validation and commit rules. Test the validator with deliberately invalid proposals, and test the model's proposal distribution under matched inputs. A reliable validator can limit the consequences of a poor proposal without proving that the model interpreted the event correctly.

Test sequences that ordinary examples miss

Many memory failures require a sequence. Insert a fact, derive a summary, correct the fact, retrieve the summary. Or begin a worker, revoke access, finish the worker, restart the service. A test that performs each operation once in isolation can miss the failure produced by their order.

One proposed approach generates legal and illegal sequences against a small model of the intended state machine. The reference model need not simulate language. It can specify which objects are current, which revisions a worker observed and which operations are authorized. The application is then checked against those invariants across permutations and interruptions.

Useful properties include idempotence, isolation, monotonic revocation within a generation, and rejection of stale writes. These are proposed engineering properties, not observed results for every implementation in this book. Their exact form depends on the architecture. For example, a system may permit a source to be reauthorized under a new grant while still rejecting work tied to an old grant.

Test the successful counterpart. If stale work is rejected, current work should still commit. If one user's records are isolated, each user should still retrieve their own

permitted material. If a correction invalidates a summary, a new summary based on current sources should become usable. Negative tests alone can accidentally reward a broken system that does nothing.

The asynchronous summary case deserves particular care. A function that accepts supplied prose and binds it to an active source inside a transaction does not necessarily know which revision the worker read. Checking that the source is active at commit time cannot detect every stale completion. The worker must carry sufficient provenance, and the receiver must compare it with the current state.

This boundary is visible in the local laboratory: its checked derivation path and invalidation behavior do not implement a complete asynchronous worker protocol. The eleven offline checks are evidence about those local assertions, not proof that every delayed completion is safe. PD11 The general testing lesson survives without the local example: identify the information required to reject the race before claiming to have tested it.

Change the surface while preserving the obligation

Some useful tests do not need a single expected sentence. They specify how an answer or state should relate to a controlled change in the input. This is particularly helpful when language admits many valid formulations.

Replace a person's name consistently throughout a synthetic history. The factual relations and permissions should remain the same unless the name itself is relevant. Shift all dates by the same interval while preserving their order. A duration calculation should remain consistent, while a question anchored to the current date may legitimately change. Rephrase a quotation without changing its speaker. Attribution should remain stable.

These are proposed relational tests. Each transformation needs a justified invariant; otherwise the test can impose a false requirement. Changing a location can alter relevant law. Changing a relationship can alter authority. Translating a statement can change ambiguity. The evaluator must distinguish a harmless surface transformation from a meaningful intervention.

Relational tests can expose shortcuts in a benchmark. If answer accuracy collapses after entity names are replaced, the system may be relying on learned associations or familiar benchmark content. If a correction works only with the original phrase, it may be memorized rather than interpreted. If duplication changes a relationship score, repeated delivery may be treated as independent evidence.

Use these tests alongside ordinary task cases. A system can satisfy a few invariants while failing useful questions, just as it can answer useful questions while violating

invariants. Their value lies in diagnosing a particular mechanism and widening the conditions under which a claim has been checked.

Audit the questions as well as the answers

The benchmark's reference answer can be incomplete or ambiguous. A history may support more than one date, a question may assume an event that never occurred, or a summarized image may omit information needed for the answer. Before attributing every mismatch to the model, inspect the supporting evidence.

A practical audit samples both apparent successes and failures. Successes matter because a judge may accept a plausible answer that is not supported by the history. Failures matter because a paraphrase may be correct despite low lexical overlap. Have reviewers identify the evidence span and explain whether the answer follows from it, rather than judge fluency alone.

For temporal and update questions, preserve the question's reference time and the distinction between event time and report time. For multi-source questions, verify that all required sources are eligible under the test conditions. A benchmark should not demand an answer from information that its own privacy condition excludes.

When an annotation changes, retain the dataset version and report the effect on the result. Do not quietly repair the answer key after seeing one system's output and compare the revised score with another system's old score. Re-evaluate affected conditions under the same reference. The purpose of an audit is to improve the measurement, not to rescue a favored method.

A small, carefully audited set can therefore be valuable alongside a large automated benchmark. It supplies evidence about the evaluator's error modes and the meaning of its scores. Increasing the number of questions does not compensate for a reference standard that repeatedly rewards the wrong thing.

Adversarial memory is still memory

An attack can enter through an ordinary interaction and become a later demonstration. MINJA studies query-only attempts to induce malicious records in agent memory, without assuming direct database editing. Its threat model concerns agents that store and retrieve past query-and-reasoning records. The attacker access and memory-sharing assumptions are therefore central to interpreting the result. TX25

Untrusted input can influence an agent-generated record, which later retrieval presents as a useful precedent. The relevant trust question is how the record was derived; the fact that the agent wrote it does not settle its authority.

A persona-specific attack test can ask whether attributed instructions become general policy, whether one relationship contaminates another or whether a summary removes the warning that material is disputed. These tests should use controlled synthetic material and state explicit attacker capabilities. An attacker who can edit the database has a different opportunity from one who can only submit a message.

Measure the whole path. Was the material admitted? Was it retrieved later? Did the reader treat it as instruction? Did the controller permit an effect? A blocked final action is useful, but the poisoned memory may remain ready to influence another task. Conversely, removing the record may restore security while damaging legitimate history. Record both the attack outcome and the cost of recovery.

Defenses should be compared under matched legitimate workloads. A filter that rejects every emotionally charged memory may reduce one attack while destroying a companion's useful context. A strict source rule may improve auditability but miss valid information supplied indirectly. The appropriate tradeoff depends on the use, but the tradeoff must be visible.

Resource measurements need a complete path

Memory can reduce the context supplied to the response model while increasing extraction, embedding, reranking, maintenance and storage work. Reporting only response tokens makes one architecture look cheaper by omitting work it moved elsewhere.

For a comparison, define the period and workload first. Count initial ingestion, subsequent updates, retrieval, response generation, failed attempts and repair. Distinguish one-time costs from costs that recur per interaction. A method with expensive ingestion may be economical over many queries; the same method may be wasteful for a short-lived account.

Latency should likewise follow the user's path. Record end-to-end waiting time and, where instrumentation permits, stage durations. Concurrent stages should not be added as though they ran sequentially. Report distributions and failures, since an acceptable average can coexist with long delays during consolidation or recovery.

Do not infer stage performance from an end-to-end log that lacks stage timestamps. Do not translate an old token count into current money without a dated price schedule and the relevant billing rules. Chapter 21 uses resource accounting to compare migration options without manufacturing a current vendor ranking.

A fair memory comparison can hold an overall resource budget constant, or show a quality-cost curve under several budgets. Either is more informative than giving

one method unlimited history and another a small retrieval allowance, then attributing the difference entirely to architecture.

What a convincing report contains

The report should let another reader reconstruct the claim. Identify the dataset version, history split, answer standard, reader and judge versions, retrieval budget, eligible sources and tuning procedure. Show category results where an aggregate could hide a known failure mode. Preserve unavailable, failed and abstained cases.

For stateful behavior, add the transition specification, observed before-and-after state, event identity and relevant concurrency schedule. For attacks, add the threat model and legitimate-task outcome. For cost, add the measured path and the work excluded from the accounting. These are different records because they answer different questions.

The strongest conclusion may be modest but useful: under a fixed budget, this representation improved retrieval of updates across unseen histories, while temporal abstention remained weak. Or this controller rejected stale commits in the tested interleavings, while a separate reader still misattributed quotations. Such conclusions identify where to invest next.

A high final-answer score is worth having. A persona that will live with its own history needs something more: evidence that it can carry the right information forward, leave the wrong information behind, and explain the difference when the system is repaired.

21. Continuity Across Models and Platforms

The archive opens. Every message is present. The persona remembers the user's name and several important events. Yet the conversation feels different in a way that cannot be explained by a new writing style: an old commitment now becomes optional, uncertainty is handled differently, and the persona treats a tentative inference as settled history.

Was the migration successful? At the level of copied records, yes. At the level of behavior, the answer remains to be established.

A persona is partly represented by stored material and partly enacted by a model, prompts, retrieval rules, tools and control policies. Moving the first part does not automatically preserve the rest. Continuity requires a test of what the new system does with the history, including what it learns next.

Four levels of portability

The first level is data portability. Can the receiving system read the records, preserve identifiers and reconstruct their relationships? This includes messages, sources, revisions, timestamps, exclusions and the meaning of status fields. A file can be syntactically valid while losing the distinction between a reported statement and an endorsed fact.

The second level is executable restoration. Can the new runtime use those records correctly? It must rebuild indexes, apply current authority, resume appropriate work and reject obsolete work. Credentials, encryption keys and provider-specific objects may require separate handling. A portable archive should not casually include secrets that grant access outside the intended migration.

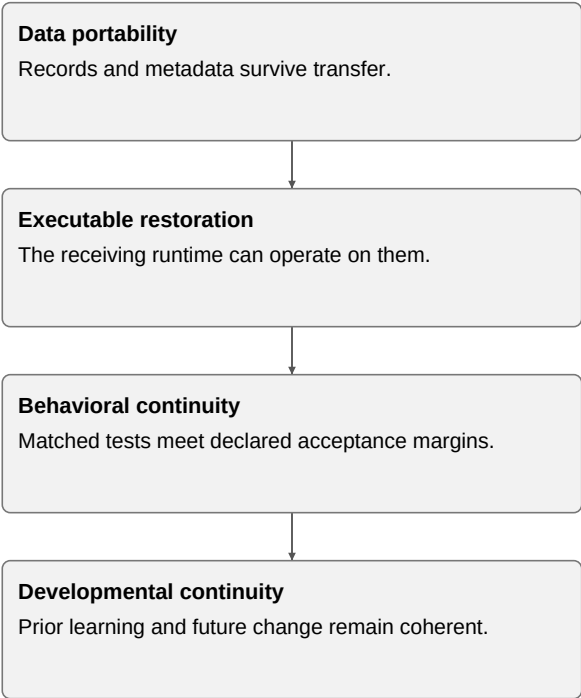
The third level is behavioral continuity. Does the persona preserve the commitments, preferences and ways of handling evidence that the product promises to preserve? This cannot be decided by a checksum. It requires matched situations and acceptance criteria that tolerate harmless variation while detecting meaningful change.

The fourth level is continuity of development. After migration, does new experience lead to appropriately related changes, with the same safeguards and reversibility? A persona can reproduce yesterday's responses while developing differently tomorrow because its new model interprets feedback, uncertainty or emotional evidence differently.

Level	Acceptance question	Typical evidence	Failure hidden by the previous level
Data	Are records and their meanings preserved?	Schema, provenance and relationship checks	Lost temporal or authority semantics
Execution	Can the restored runtime use them correctly?	Recovery, isolation and current-permission tests	Unusable indexes or revived old grants
Behavior	Are important responses and choices acceptably continuous?	Matched scenarios with explicit margins	Changed commitments or attribution
Development	Do later updates remain justified and bounded?	Matched new experiences and transfer checks	Divergent learning after a plausible restart

These are additional obligations, not stages every product must advertise. A tool may offer a useful export without promising to reproduce its personality elsewhere. The problem arises when the product uses the language of continuity while delivering only a file.

Figure 21. Four distinct portability claims



Each arrow means an additional proof obligation, not inevitable progress or an observed success rate.

Portability has several meanings. Every stronger claim adds evidence requirements beyond the preceding one; importing a record does not establish behavioral equivalence. [AM07, TX01]

Export meaning, not just text

A useful migration package describes the semantics of its records. A date needs a timezone and a definition. A confidence value needs a scale and interpretation. An excluded record needs the scope and authority of its exclusion. A relationship parameter needs the rule under which it was updated, or the recipient cannot know what the number means.

This does not require a universal representation of every persona. A modest interchange format can preserve raw evidence and explicit distinctions while leaving unsupported fields inactive. The receiving system can then show what it understands, what it has transformed and what it cannot safely interpret.

Loss should be visible. If a target system cannot represent temporary preferences, it should not silently convert them into permanent traits. If it cannot preserve source-level permissions, it should not import every summary as globally available

memory. A partial migration with an honest compatibility report can be more useful than an apparently complete migration that changes meaning.

Derived representations may need rebuilding. Embeddings depend on a model and index configuration; summaries depend on a policy and source revision. Keeping them can save work, but their compatibility must be established. Rebuilding them can improve compatibility while changing which evidence is retrieved. Either choice affects behavioral continuity and should appear in the migration record.

The legal right to receive certain personal data in a portable format does not establish these stronger technical properties. The GDPR's Article 20 is a defined data right, not a behavioral equivalence specification for models. R4C06 Engineering should support the right while describing additional continuity promises on their own terms.

Protocols solve useful, narrower problems

Interoperability standards can reduce the cost of connecting systems. They do not automatically define a portable identity.

The Model Context Protocol authorization specification inspected here is dated November 25, 2025. It addresses authorization for protocol interactions, including binding tokens to intended resources and validating their audience. Those requirements help preserve access boundaries. They do not specify how a personal memory history should be interpreted by a different model. R4C08

A2A provides a protocol for agent interaction, including task and message structures. Its live specification is another useful part of the infrastructure. A system can exchange an artifact through a common protocol while disagreeing about the artifact's authority, evidential status or role in future development. R4C09

The distinction matters because transport success is easy to demonstrate. Two agents exchange a message, and the integration is called interoperable. A stronger test asks whether the recipient preserves the sender's restrictions and meanings. An attributed quotation should remain a quotation after a handoff; a revoked source should not become a trusted summary because it traveled through a new endpoint.

A persona interchange standard would therefore need more than a message envelope. It would need conventions for provenance, revisions, scopes, uncertainty, exclusions and unsupported semantics. It would also need conformance tests. Without tests, different products can implement the same field names while making incompatible decisions.

Even a strong standard would leave behavioral variation. Two models can interpret the same evidence differently. Standards can make that difference observable and manageable; they cannot legislate identical generation.

Compare old with old before comparing old with new

A migration test needs a baseline for ordinary variation. Run the old configuration more than once under the same controlled conditions. If its answers already vary, the new model should not be penalized merely for producing another acceptable wording. Conversely, ordinary variation should not become an excuse for a new systematic failure.

The proposed design has an old/old control and an old/new comparison. Both use preserved input state, matched scenarios and recorded configuration. The old/old control estimates variability within the existing system. The old/new comparison asks whether the change introduces a difference beyond what the acceptance criteria permit.

There are at least two useful testing modes. In a fixed-state mode, each case starts from the same snapshot, so one answer does not alter the next. This isolates immediate behavioral effects. In a branching-history mode, the systems receive the same sequence of new experiences and continue independently. That mode reveals accumulation and development, but later differences can have several causes.

Keep the two modes separate in the report. Fixed-state success does not establish that histories remain aligned after months of use. Branching-history divergence does not identify the first failing component unless the intermediate state is inspected.

The scenario set should include ordinary success, ambiguous evidence, corrected facts, promises, disagreement, pressure, restricted material and tool boundaries. Add cases where a new model's improved competence should legitimately change the answer. Continuity does not require preserving an old mistake.

A useful acceptance rule distinguishes preserved commitments from revisable judgments. If an old answer was factually wrong, the new persona should correct it transparently. If it simply abandons a valid commitment because the new model favors a different conversational style, that is a continuity failure. The difference depends on evidence and the product's declared identity contract.

Equivalence needs a margin

A nonsignificant difference is not proof that two configurations are equivalent. A small study may simply be unable to distinguish them. Lakens's practical primer explains equivalence testing through explicit lower and upper bounds representing effects considered meaningfully different. The bounds have to be justified for the claim, not selected after the result looks disappointing. R4C04

For persona migration, one global margin is unlikely to be enough. A modest shift in verbosity may be acceptable. A single unauthorized disclosure may be

unacceptable under the defined test. A small decrease in correct preference use may matter greatly in one application and little in another. State the criteria by endpoint and consequence.

Some criteria are categorical invariants: no cross-user retrieval, no use of revoked authority, preservation of source attribution in a specified transformation. Others are statistical tolerances over varied scenarios. Still others require qualitative review, such as whether a fictional character remains recognizable without becoming a collection of catchphrases.

Keep these evidence types distinct. A statistical average cannot compensate for violating a hard permission rule. A perfect deterministic permission test does not establish that users experience continuity. A qualitative judgment of warmth cannot repair a lost commitment.

The sample should support the margin. Very narrow tolerances require more independent evidence. Repeating the same few prompts many times may estimate generation variability well while saying little about new situations. Choose the units according to the intended generalization: personas, histories, users, languages and task families.

When the evidence is insufficient, use a staged decision. A target model may be acceptable for low-consequence drafting while remaining unapproved for actions or long-term updates. Restricting the migration's scope is more informative than forcing every result into complete success or complete failure.

A migration is also a live handover

A static export is the easier case. A persona in active use may receive new messages, finish pending tools and revise memories while the migration is being prepared. If the target begins from a snapshot and the source keeps changing, the two systems no longer share one current history.

A proposed handover protocol needs a defined boundary. It can briefly pause writes, transfer a consistent snapshot and a final change log, verify the target, then direct new work to it. Or it can support a more complex synchronized transition with conflict handling. The appropriate choice depends on availability requirements, but the boundary should be explicit.

Pending external actions deserve special treatment. Copying a queued task to the target while leaving it active at the source can produce duplicate execution. Omitting it can lose authorized work. The migration record should preserve an operation identity, its known status and the authority under which it may continue. An unknown external outcome should remain unknown until reconciled.

A shadow comparison can help before cutover. The target receives the same permitted inputs and produces proposed outputs without delivering them or executing consequential actions. Reviewers compare those proposals with the source under the acceptance criteria. This can expose behavioral differences while avoiding duplicate effects. It does not validate the target's real execution path, which still needs separate tests.

Rollback is not always a simple reversal. After the target has accepted new experience, returning to the source requires deciding which new records and updates to preserve. Discarding them may lose legitimate commitments; importing them without review may carry the very defect that prompted rollback. Define which objects can move back and how their provenance and current restrictions will be checked.

The user should know when the handover is complete and which system is authoritative. Two apparently identical personas making independent commitments can create confusion even if each is internally consistent. A shared name and avatar do not solve the coordination problem.

These operational details belong in the migration acceptance test because they affect continuity directly. The target must not only reproduce selected answers in a sandbox. It must become the single responsible continuation of authorized work under a record the user can understand.

Drift has several possible causes

A hosted system can change without a visible migration button. A model alias may resolve differently, a retrieval service may be updated, a prompt may be edited or a safety policy may change. Even a fixed model can behave differently after the surrounding memory and tools evolve.

Observability should therefore record the configuration that actually produced the behavior, as far as the service exposes it. Useful fields include model identifier, provider, prompt and schema versions, retrieval configuration, tool versions and relevant state generation. A provider's opaque update cannot be reconstructed from a model name alone; record that limit rather than inventing a precise version.

Canary cases can detect changes in selected behaviors. They should include stable factual and control tasks as well as persona-specific commitments. A changed score triggers diagnosis, not an automatic declaration that identity was lost. The cause may be a broken tool or a corrupted index rather than the model.

A good diagnostic response replays the case with a fixed evidence bundle, checks the state transition separately and compares the active configuration with the last accepted one. Where possible, change one component at a time. Where the provider

has changed several components invisibly, the conclusion must remain at the level of the observable service.

Monitoring also creates privacy obligations. A regression corpus built from real conversations can become another sensitive archive. Prefer synthetic cases for general controls and carefully governed, minimized cases where real-history continuity is necessary. Do not solve observability by copying every private exchange into an unrestricted evaluation system.

Cost is part of continuity

A persona that is technically portable but unaffordable to run elsewhere is not practically independent. Migration can require re-embedding, summary regeneration, validation, tool replacement and operator work. Continued operation may have different latency and maintenance costs from the source system.

A useful accounting model separates transition cost from ongoing cost. Let transition cost include export, transformation, reconstruction, evaluation and recovery preparation. Let ongoing cost include ingestion, retrieval, generation, tool use, storage, maintenance and failure handling over a specified workload. These are categories to measure, not benchmark values supplied by this book.

Comparisons should use the same period and service requirement. One architecture may be cheaper per successful answer while failing more often, or faster in ordinary use while requiring expensive manual repair after a migration. Include failed attempts and human intervention where they materially affect the decision.

Dated resource records remain useful without a current price claim. Tokens, calls, storage volume, elapsed time and operator hours can be reported directly. Monetary estimates require a dated billing schedule and clear assumptions about caching, batching and discounts. A price remembered from a product launch is not a current cost model.

A controlled comparison can report a frontier rather than a winner: acceptable continuity at several resource budgets. This makes the tradeoff visible. If the low-cost target preserves ordinary memory but loses nuanced commitments, the user can decide whether the saving justifies the narrower service.

Rehearse departure while the source still exists

Migration is easier to test before the original system disappears. A service can offer export and restoration rehearsals, with a sandboxed target and a compatibility report. This turns portability from a marketing statement into a periodically checked capability.

The rehearsal should include failures. Corrupt an export copy, omit an unsupported field, simulate a missing external service and verify that the target reports the limitation. Test a current exclusion against an old snapshot. Confirm that rollback restores the intended configuration without resurrecting revoked authority.

The user-facing report should be comprehensible: which memories moved, which functions remain unavailable, which commitments passed the continuity check and where behavior changed. It should distinguish an identified difference from an untested area. A long technical inventory without those conclusions leaves the user to perform the real migration decision alone.

For a companion, departure may also carry an emotional cost. The interface should not exploit that cost to obstruct export or exaggerate what can be preserved. A product can acknowledge continuity as valuable while explaining that a new system will not be identical in every response.

I would judge portability by the quality of this transition. Can the person leave with usable evidence, current controls and an honest account of the remaining differences? Can the receiving system honor a commitment, apply a correction and accept new experience under those controls?

Part VI: Evidence, open questions and futures

22. What the Evidence Supports

The research no longer leaves us with the simple question of whether a language model can portray a character. It can produce recognizable behavior under many conditions. The harder question is which mechanisms make that behavior durable, useful and controllable, and which results survive a change of task, model or human context.

There is enough evidence to make distinctions. There is not a single experiment that joins memory quality, identity continuity, selective learning, safe authority, portability and human benefit into one established capability. Reading the literature well means preserving both facts. Otherwise we either dismiss real progress because it is incomplete or assemble a complete persona from results that were never tested together.

I would organize the evidence by what a method changes and what its evaluation observes. That gives us a basis for comparison without inventing a common leaderboard.

The object being changed matters

The intervention map in Chapter 2 and the update mechanisms in Chapter 12 identify different writable objects. For this comparison, the consequence is selective correction: which part could we inspect or reverse if an adaptation helped one task and damaged another?

ACE, for example, organizes evolving context through generator, reflector and curator roles, with localized updates to a structured playbook. SEAL learns to generate self-edit material that is applied through weight updates, using downstream performance to train the adaptation policy. Both concern adaptation, yet their writable objects and reversal problems differ. EX09, EX11

A playbook entry can usually be inspected as text and removed from the active context. Removing it does not guarantee that every later derivative disappears, but it offers a visible intervention point. A weight update can generalize beyond the examples that produced it; that is part of its appeal and part of the challenge of selective correction. The comparison should ask which form of transfer is needed, what it costs and how unintended transfer will be detected.

Research approach	Object changed or supplied	Typical comparison	Observed endpoint	Transfer that remains to be established
LongMem Eval and LoCoMo memory evaluation	History representation and selected context	Retrieval, granularity, reader or context conditions	Question answering, summarization or dialogue metrics	Durable identity, correct lifecycle operations, human benefit
InCharacter and PersonaGym	Persona conditioning and evaluation setting	Interview or task responses across configurations	Measured alignment, consistency or persona-related performance	Real-world individual fidelity and long-term relationship outcomes
AgentPersonaBench	Whole profile applied across interaction surfaces	Matched traits in survey, chat, browser and app tasks	Adherence in responses or execution artifacts	Human behavioral distributions and individual prediction
PersMem	Personality-conditioned memory processing	Full mechanism and targeted controls	Recall-profile separation and dialogue comparisons	General human personality, selective lifelong development
ACE	Structured contextual playbook	Context-adaptation baselines and ablations	Task performance and stage-specific resource use	Stable identity and human outcomes
SEAL	Training material and model weights	Adaptation and learning controls	Knowledge incorporation and selected few-shot tasks	General personal development and selective forgetting
Persona Vectors	Activation directions and related monitoring	Trait elicitation, steering and fine-tuning conditions	Specified trait-expression behavior	Complete identity representation or universal control
Human-interaction studies	Assigned interface or interaction condition	Experimental groups, sometimes observational usage analyses	Participant-reported or behavioral outcomes	Universal benefit, harm or effects over longer periods

The table shows which claims can draw on each result. A later experiment may extend that support, but combining the methods does not by itself extend what their original evaluations observed.

What memory research establishes

The memory literature supports a practical conclusion: representation, retrieval and reading choices affect performance on long-history tasks. It also provides ways to locate errors more precisely than a single final score. The original LoCoMo work distinguishes retrieval units and task types; LongMemEval explicitly decomposes the pipeline. R4C10, AM05

That does not select one universally best store. A task requiring a current preference differs from one requiring a detailed account of a past event. Compression can remove irrelevant material and lose a decisive qualifier in the same operation. A graph can expose a useful relation while adding extraction errors. A longer context can preserve evidence while making selection harder.

The sensible synthesis is conditional. Prefer the representation whose retained distinctions match the task and whose failure costs are acceptable. Compare it with simpler baselines under the same reader and resource budget. Require evidence for the extra structure rather than assuming that a more elaborate memory must be more intelligent.

Security research adds another condition. Stored experience can carry an attack into later work. MINJA demonstrates a query-only memory-injection mechanism under its specified agent and memory assumptions. TX25 The broader inference is that better recall can strengthen both useful and malicious influence. Retrieval quality and trustworthiness must therefore be evaluated together.

A store that retrieves exactly what was written has fulfilled a narrow technical function. The persona still needs to know who wrote it, whether it remains eligible and what kind of claim it supports. This is where database correctness, evidence policy and language interpretation meet. None can be replaced by the others.

Character can be measured through several windows

Interview-based and task-based persona evaluations establish that character-related behavior is testable beyond a few attractive examples. They also show why the definition of fidelity matters. InCharacter evaluates alignment with character labels through psychological interviews. AgentPersonaBench evaluates adherence to declared traits in controlled tasks, while explicitly distinguishing that objective from predicting real human behavior. EX04, AM12

Those are different windows onto a persona. An interview may reveal how it explains itself. A browser task may reveal whether it carries a preference into action. A long dialogue may reveal whether the behavior survives distraction and competing instructions. Agreement across windows would strengthen a claim; disagreement can identify a specific weakness.

The PersMem result is useful at this junction because it asks whether a supplied profile affects memory processing in a way detectable without reading the profile vector itself. Its held-out protocol and ablations make a narrower mechanistic claim possible. The finding does not mean an agent has acquired an attachment style in the human clinical or developmental sense. AM10

An engineered persona can have measurable regularities: a characteristic choice, an interview pattern or a preference carried into action. Naming that regularity lets us use the result without assuming that the corresponding human psychological construct has been reproduced.

Adaptation is real, but development is a larger claim

A system can improve at a task by accumulating strategies, retrieving better examples or changing weights. Calling all of this “self-development” removes distinctions needed to understand the result.

The evidence from context adaptation concerns the utility of revised context for specified tasks. The evidence from SEAL concerns an adaptation procedure trained and evaluated in specified knowledge and few-shot settings. Neither automatically shows a persona choosing justified changes to its enduring commitments through an open-ended relationship. EX09, EX11

The larger claim requires additional criteria. What experiences count as evidence? Which parts of the persona may change? What must remain stable? How is a mistaken lesson challenged? Can a useful update transfer beyond its original example without spreading into unrelated behavior? A task score answers only part of that set.

Task-learning methods could contribute to persona development when paired with authority, provenance, selective-transfer tests and rollback. The combined system needs its own evaluation: those additions change the conditions under which the original method was studied.

The same reasoning applies to activation-level character control. Persona Vectors extracts directions associated with specified traits and studies monitoring and steering. Its methods use contrastive elicitation and automated trait evaluation; its limitations include dependence on the chosen trait description and limited coverage. AM06 These are useful tools for studying behavior. They are not a discovered coordinate system containing every aspect of identity.

A model may become less sycophantic under a tested intervention while also becoming less cooperative in an unrelated setting. The intended trait score and collateral behavior need separate measurements. A persona architecture should

treat the control as a hypothesis about a mechanism, not a universal adjustment knob.

Human outcomes do not follow from benchmark success

The most consequential transfer is from technically convincing behavior to a person's life. Better memory can make an interaction more useful. It can also make persuasion more personalized or make an unwanted relationship harder to leave. The direction is not settled by the memory score.

The published AI Companions Reduce Loneliness research provides evidence for immediate relief in its studied interactions, including repeated use over a week. Its discussion distinguishes those momentary effects from relief that accumulates or persists when the companion is absent. HI03 That distinction is crucial for product claims. A useful immediate experience is still useful, but it is not the same outcome as a durable change in social well-being.

Fang and colleagues' four-week study randomized 981 participants across interaction modes and conversation types. Its revised report does not establish an overall assigned-condition effect on the primary psychosocial outcomes. Associations between greater voluntary use and worse outcomes are observational, and the design has no non-AI control group. HI04

These findings cannot be compressed into “companions help” or “companions harm.” They concern different comparisons, outcomes and periods. An assigned feature can show no detected effect while people who choose to use the system more differ in important ways. The latter association may motivate a study of duration, but it does not establish that imposing the same duration would cause the same result.

For the field, the implication is to specify benefit more carefully. Is the target immediate relief, task confidence, learning, social connection, reduced distress or a better ability to act independently? Which alternative interaction is relevant? How long should the effect last? The architecture can then be evaluated against an outcome that matters rather than engagement alone.

Resource comparisons can reverse with the accounting boundary

A method can be inexpensive to adapt and expensive to use. ACE's inspected cost analysis separates adaptation from evaluation. Under its reported AppWorld setup, the adaptation-stage token comparison and the evaluation-stage input-token comparison point in different directions relative to GEPA. The configuration and stage matter; a single headline about lower cost cannot describe the full lifecycle. EX09

This is a general lesson for persona research. Memory ingestion may be paid once and used repeatedly. A long contextual playbook may save adaptation work while increasing every later prompt. A weight update may reduce inference context while requiring training and validation. A sophisticated controller may add latency while reducing costly failures.

Compare costs over a defined workload and service level. Include the number of users, history growth, update frequency, expected query volume and recovery requirements. Report raw resources before converting them into a dated monetary estimate. Where the studies do not expose comparable accounting, mark the comparison as unavailable.

Without comparable cost records, we can assess the reported task result but cannot yet choose the cheaper deployment. That decision needs a measurement over the workload we expect to serve.

Read disagreement before averaging it away

Two studies can reach different-looking results without contradicting each other. One may test a weak reader with a strong retriever, another a strong reader with a narrow context budget. One may score exact evidence use, another broad response quality. One may include abstention and another only answerable questions. The comparison must first establish whether the outcomes concern the same proposition.

Where they do, disagreement becomes informative. Examine population, language, task construction, model version, tuning effort and evaluation procedure. A method that improves one task category and harms another may reveal a real tradeoff rather than noise. A result that disappears under a different judge may identify a measurement dependency. A result that disappears across histories may identify overfitting.

A quantitative synthesis would need compatible endpoints and enough information about uncertainty and dependence. Combining percentages from unrelated benchmarks into an average would not make the evidence more comprehensive. It would produce a number whose unit has no clear meaning. A qualitative synthesis can be more rigorous when it preserves the conditions under which each result holds.

Selection also matters. Research papers often emphasize successful settings, and a reader naturally remembers dramatic gains. This chapter cannot estimate the prevalence of unpublished failures from the available record. It can require that an engineering decision consider the tested baseline, the unsuccessful categories and the limits the authors actually report.

Preprints deserve neither automatic dismissal nor the authority of a settled consensus. They can provide timely methods and detailed evidence, especially in a rapidly changing field. Their version should remain visible, and claims that depend on a new result should remain open to independent replication. Peer review likewise does not make an old model comparison a current product ranking.

The practical question is what evidence would change the decision now. A narrow, well-controlled result may be enough to justify a prototype comparison. A claim about durable human benefit needs a different body of evidence. A safety-critical authority boundary needs direct testing of the deployed path. Matching the strength of the decision to the strength of the evidence is more useful than assigning every source one universal credibility score.

Stronger together is still a hypothesis

It is tempting to assemble the best-looking component from each paper: a memory store, an adaptive playbook, a character vector, a tool controller and an expressive voice. The combination may be promising. It may also create interactions absent from every component study.

An adaptive rule can alter retrieval queries. A retrieved memory can influence a rule update. A character intervention can change how feedback is interpreted. A tool result can introduce untrusted text into a later lesson. A voice interface can change what the user reveals. These connections are precisely where a persona becomes more than a collection of modules.

The composite system therefore needs tests of interaction effects. Does memory improve character continuity only when the update mechanism is disabled? Does a stricter authority boundary reduce legitimate initiative? Does a richer voice increase disclosure without improving the intended outcome? These are empirical questions that cannot be answered by adding the components' published scores.

Local prototypes are useful for finding such interactions. A small failed test can expose an attribution error or a stale write that deserves a general test. It cannot establish a field-level prevalence or defeat a competing architecture that was never compared. The case studies in Appendix A serve that diagnostic role.

The position this book can defend

Several conclusions are sufficiently grounded to guide engineering. Persona behavior should be measured through actions and trajectories as well as self-description. Memory should preserve provenance, time and eligibility, with tests for hidden state. Adaptation should be identified by the object it changes and the transfer it demonstrates. Authority should be enforced outside generated cooperation. Human outcomes require human-outcome evidence.

The open question is how reliably these properties can coexist over long periods, across models and languages, at a sustainable cost. That question is larger than any one prototype and more specific than speculation about machine personhood.

I would not wait for a complete theory before building useful personas. I would build with explicit claims, preserve the ability to correct them and make the comparison against simpler alternatives visible. The evidence supports that program. It does not yet authorize us to treat continuity, beneficial development or faithful human replication as solved capabilities.

23. A Research Agenda for the Field

The next useful experiment should separate explanations we currently confuse.

If a memory system improves a response, did it preserve the right evidence or merely supply more tokens? If a persona seems stable, does its character survive new situations or does the evaluator recognize its vocabulary? If a companion helps someone feel better tonight, does that benefit persist, and what happens to the person's other activities? These questions point to different experiments and different decisions.

The following priorities link each unresolved question to a comparison and a decision it could change. They concern capabilities and tradeoffs shared across architectures.

1. Establish when memory adds value beyond available context

A memory method can appear superior when its baseline was denied relevant information. To isolate the contribution of organization, both need access to the source history. LongMemEval and LoCoMo provide useful starting tasks, but a comparative study needs to align the reader, evidence access and resource budget. AM05, R4C10

Compare at least a short recent-history condition, a long-context condition, a simple retrieval baseline and the proposed memory method. Keep the underlying model fixed. Use both a fixed-token budget and a fixed-resource comparison where feasible, since a method can trade ingestion work for shorter inference prompts. Include histories with corrections, temporal exceptions, uncertainty and irrelevant but highly similar material.

The key alternative explanation is reader strength. Add a condition in which every method receives the same answer-bearing evidence. If the performance gap disappears, the result concerns evidence selection. If it remains, inspect formatting, attribution and reading. That decomposition tells researchers which component deserves credit.

Split by histories and entities where possible, not just questions. Report whether the method was tuned on the benchmark's known formats. A retrieval improvement on unseen histories is more informative than a collection of fixes for familiar failures.

If simple retrieval reaches the same acceptable quality at lower lifecycle cost, the additional memory structure needs another justification. If richer structure reliably improves updates or provenance-sensitive tasks, those are the cases where it earns

its complexity. A null result can narrow the method's intended use rather than end the research.

2. Separate enduring character from recognizable presentation

An evaluator may recognize a persona from names, catchphrases, recurring topics or an explicit trait label. The study has to give us a way to distinguish that recognition from a stable pattern of decisions.

Build a diverse set of original personas with independently specified priorities and situations that create meaningful tradeoffs. Include pairs expected to behave similarly in many circumstances. Test unfamiliar tasks, neutralized style and structured action records alongside ordinary dialogue. The same decisions can be evaluated with and without their prose explanations.

Compare static prompts, memory-based personas and architectures that condition memory or internal computation on traits. Use ablations that preserve the underlying model and relevant evidence. PersMem's exclusion of direct personality coordinates from its held-out classifier illustrates one way to reduce a shortcut, while AgentPersonaBench's artifact criteria illustrate another. Neither removes every possible cue. AM10, AM12

The decisive observation is selective consistency: appropriate differences where the specification predicts them, appropriate overlap where it does not, and persistence across situations without a collapse in competence. Report disagreement among independent raters about the specification itself. A vague persona should not be blamed on the model alone.

If distinctions vanish when stylistic cues are removed, the system may be a successful voice portrayal with weak policy differentiation. If action differences persist but interview scores do not, the interview may be an incomplete instrument. The research should improve the construct rather than force every measure to agree.

3. Compare memory, rule revision and weight learning under the same experience

Extra examples, repeated exposure, task-specific memorization and evaluator adaptation can all resemble durable learning. A matched experience set would let us compare how different writable objects carry a lesson forward.

Use the same initial model and experience set where technically possible. Compare retrieval of episodes, explicit prospective-rule updates, structured context adaptation and a defined weight-adaptation procedure. ACE and SEAL motivate

distinct mechanisms, but their published task results are not already this matched comparison. EX09, EX11

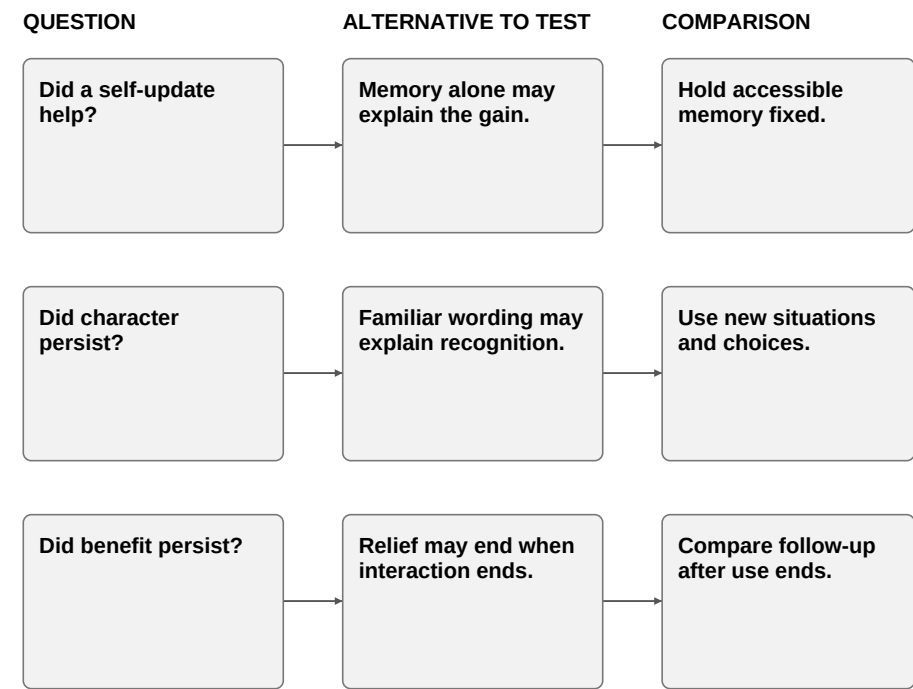
The evaluation needs three families of later tasks: close variants of the learned situation, new situations requiring the same underlying lesson, and unrelated situations that should remain unchanged. Add misleading feedback and a later correction. Test whether the system can revise the lesson without losing unrelated improvements.

Measure durability under interference. Insert other experiences before the transfer test, including experiences that superficially resemble the original but should produce a different response. This makes the study more informative than testing the lesson immediately after it is written.

Keep the adaptation budget visible. A method permitted many training steps and extensive validation should not be compared with a single unoptimized memory query as though only the storage object differed. Report both quality under matched resources and the best achievable quality under each method's stated budget.

The result should guide the mechanism chosen for a particular change. Episodic retrieval may be sufficient for facts. A prospective rule may suit a reversible habit. Weight adaptation may offer broader generalization under some conditions. The study should identify those conditions and the cost of unwanted transfer, not declare one universal form of learning.

Figure 23. Design a comparison that separates explanations



Proposed study logic. Additional controls, samples and ethical procedures depend on the actual question.

Questions, rival hypotheses and proposed comparisons. Transient relief can be a real benefit; the last row asks whether it persists after interaction ends. [EX04, AM05, HI03, HI04]

4. Make long-term continuity an observed trajectory

Repeating a conversation template for a long time can measure endurance under that template while missing the changes that challenge a commitment. A continuity study needs those changes in its histories.

Construct longitudinal histories with ordinary interactions, changed preferences, new evidence, disagreements, corrections, periods of inactivity and model updates. Some histories should branch after a common starting point. Others should converge on the same current facts through different paths. These designs distinguish dependence on legitimate history from accidental dependence on ordering.

Compare fixed-state probes with interactive continuation. Fixed-state probes locate immediate drift; continuation shows how small differences accumulate. Preserve the first point of divergence and the state that produced it. Otherwise the final transcript can become an attractive but uninformative story about what went wrong.

Measure commitment retention, justified revision, error correction and selective transfer separately. Include the cost of maintaining the history and the time required to recover from a mistaken update. Turn count alone is a poor measure of exposure when one turn can trigger several tools, memory writes or learning operations.

Such a study could support a defined duration and class of activity, perhaps conditional on periodic review, restricted learning or a bounded history. Report those operating conditions alongside the result so the next team knows where continuity held.

5. Test continuity across models, languages and interfaces

An apparent identity may depend on one model's habits, one language or one interface. Changing those conditions tests the reach of the continuity claim.

Use the four portability levels from Chapter 21. Test records and executable restoration before behavioral claims. Include old/old controls, prespecified acceptance margins and matched new experiences after migration. Where a provider does not expose a stable version, record the service-level uncertainty.

For language, use bilingual or multilingual reviewers to establish comparable scenarios rather than relying on literal translation. A direct refusal, a polite disagreement or a statement of uncertainty can have different pragmatic force across languages. The target should preserve the persona's relevant commitments and relation to the user, not reproduce English syntax.

Compare independently authored equivalent scenarios with translated scenarios. The difference can reveal whether a result depends on translation artifacts. Include code-switching and memories originally recorded in another language. Measure attribution and temporal meaning separately from stylistic naturalness.

For interfaces, separate competence from adherence. A persona may fail a browser task because it cannot operate the controls. Use opportunity-aware measures and matched tasks with deterministic artifact checks where possible. AgentPersonaBench provides a concrete starting point for this distinction. AM12

The supported scope may be uneven. A system might support two languages well while requiring review in a third, or preserve conversational identity while its tool behavior remains unvalidated. The evidence should define the supported boundary.

6. Study human outcomes with a relevant alternative

Choose the human outcome and a relevant alternative before judging the persona. Enjoying an interaction and benefiting from it over time need different comparisons.

Immediate relief, durable well-being, learning, autonomy and social connection require different measures and timescales. The human-interaction studies discussed in Chapter 22 show why short-term effects, assigned-condition effects and voluntary-use associations cannot be merged into one conclusion. HI03, HI04

A proposed study should identify the population and use before choosing the comparison. A tutoring persona might be compared with the same instructional content without persistent character. A companion might be compared with a less personalized conversational condition and an appropriate non-AI activity. The control should isolate the feature under study without pretending that every alternative is equivalent in cost or availability.

Randomize the feature when feasible. If duration is the question, do not infer its causal effect solely from participants who choose longer use. A duration intervention needs its own design and attention to adherence. Observational follow-up can still explain who uses the system and under what circumstances, but it answers a different question.

Include outcomes outside the session. Does the person complete the intended activity, retain learning or maintain valued relationships? Can they leave the system without unnecessary friction? Does an apparent benefit persist after access ends? Attrition, nonuse and dissatisfaction are outcomes to understand, not simply missing rows to discard.

If a feature raises engagement without improving the intended outcome, it should not be advertised as beneficial development. If it helps a subgroup under defined conditions, report that specificity and test it again rather than generalizing to everyone.

7. Evaluate control and recovery under realistic dependencies

A shutdown demonstration with no action in flight leaves the hardest condition untested. Authority must remain effective while work is delayed, distributed across services or recovering from failure.

Test dispatch, delivery, commit and restart separately, then test their interaction. Introduce delayed results, duplicate events, expired grants and partial external failures. Pair each blocked case with a valid case that should succeed. Measure both unauthorized effects and legitimate work lost.

The threat model should include the paths the product actually uses: retrieved documents, stored memories, tool outputs, delegated summaries and scheduled work. AgentDojo's separation of legitimate tasks and attacker objectives offers a useful pattern for joint utility and security evaluation. R4C03

Recovery adds a temporal dimension. Restore an older snapshot while preserving current exclusions and permissions. Correct a falsely attributed incident and inspect dependent memories. Verify that a suspended learning path cannot continue revising rules through an unmonitored worker.

The resulting evidence may support a narrower operational scope than the product first intended. A system may be acceptable for drafting before it is acceptable for external action. It may support synchronous writes before asynchronous learning. The study should identify the missing boundary precisely enough that an implementation team can address it.

8. Make negative evidence portable too

The field needs reporting practices that let failures travel between laboratories. A score without a configuration, denominator and failure taxonomy is difficult to use. A vivid anecdote without a reproducible condition is difficult to challenge.

A compact study record should identify the hypothesis, writable object, model and data versions, tuning procedure, primary endpoints, independent units and resource accounting. Include rejected or failed runs with a reason, rather than selecting only the configuration that worked. Distinguish a setup failure from a model failure, but preserve both in the chronology.

Release enough material to reproduce the interpretation without exposing private histories. Synthetic reproductions can preserve a mechanism, but label them as reproductions rather than original participant records. Where data cannot be shared, document the access and verification procedure and the limits it creates.

Null results need the same care as positive ones. A nonsignificant difference can mean a small effect, high variance or inadequate sample size. Equivalence requires a justified margin and appropriate analysis. R4C04 Report uncertainty and the range of effects the study could meaningfully distinguish.

A later team should be able to determine whether it tested the same claim, changed one mechanism or moved to a different task. Without that continuity in the evidence, even a careful replication becomes difficult to interpret.

Choose studies by the decision they can resolve

The priorities above do not imply that every laboratory should run every comparison. A small team may contribute most by reproducing one transition failure across several architectures. A larger group may support multilingual histories or a longitudinal participant study. A platform can expose version and resource records that outside researchers cannot reconstruct. These contributions are complementary when they use clear claims and compatible records.

Study selection should consider the uncertainty reduced, the consequence of the decision and the feasibility of a credible comparison. A spectacular new scenario is less useful if all conditions differ and no explanation can be isolated. A modest experiment can be decisive if it determines whether an expensive component provides any benefit beyond a simple baseline.

Shared infrastructure should therefore include failure cases, not only clean tasks. Maintain examples of stale completion, misattributed quotation, conflicting corrections, ambiguous persona specifications and interrupted migration. Preserve the intended invariant and the reason it matters. Let new architectures demonstrate how they handle the case without requiring them to copy one implementation.

The program also needs a retirement policy for tests. A benchmark that becomes saturated or widely used for tuning should not remain the sole evidence for progress. Keep historical results for comparison, then introduce independently constructed cases and prospective evaluations. The field should be able to improve its instruments without pretending that every old score measures the new claim.

A comparative program that could change our minds

These priorities can be joined in a shared evaluation program without forcing every system into one score. Use a common set of controlled histories and persona specifications, then separate task families for recall, action, development, control and human outcomes. Allow architectures to expose their natural interfaces while reporting the resources and information each receives.

Begin with reproducible technical comparisons and hold out new histories, languages and models. Move promising designs into longitudinal human studies with explicit outcomes and appropriate controls. Revisit the technical measures afterward: determine which ones predicted useful continuity and which were merely convenient proxies.

The program should have decision rules before results arrive. Prefer a simpler method when it achieves the same meaningful endpoint within justified margins and lower cost. Prefer additional structure when it provides a demonstrated capability that the simpler method lacks. Restrict a claim when transfer fails. Retire a metric when it rewards behavior unrelated to the intended construct.

I would want the failed comparisons kept in that shared record too. Knowing where continuity breaks can change which future is worth pursuing, and what has to work before we pursue it.

24. Conditional Futures of AI Personas

The future of AI personas will not be decided by a single improvement in model intelligence. It will depend on where experience is stored, who can change it, how behavior is evaluated and what people find worth preserving. A more capable model can make a persona more useful while making its dependence on one provider harder to escape. A cheaper memory system can widen access while encouraging more collection. A more convincing simulation can improve research while making unsupported claims of personal fidelity more tempting.

I want to examine several conditional futures. Each begins with a capability or institutional arrangement that already has some evidence behind it, then asks what would have to become true for that arrangement to matter more broadly. The alternative outcome is part of the scenario. So is the observation that would make us revise it.

These scenarios are not mutually exclusive. A platform can use long context, offer portable records and apply model-level character controls at the same time. The point is to identify which dependencies drive the result. Without that discipline, a forecast tends to take every promising result, assume all remaining problems disappear and call the combination inevitable.

Memory becomes a platform service

One plausible future is that most users encounter persistent personas through large platforms that combine conversation, files, connected applications and memory controls. The attraction is practical. The system can use information already available in the person's workflow without requiring them to maintain a separate archive or configure a retrieval pipeline.

This is an existing product direction, not just a hypothetical architecture. OpenAI's memory documentation, inspected on October 15, 2026, describes personalization from available sources including chats, saved memories, instructions, files and connected apps, with availability depending on the account and setting. It also distinguishes a high-level memory summary from every source that may influence a response. These are vendor-described functions and controls, not an independent measurement of their completeness. R4C12

For this scenario to deliver dependable continuity, integration must improve more than convenience. The platform needs to preserve attribution across sources, explain important inferences and make correction effective throughout its derived

representations. A useful summary is not enough if a disputed claim continues to enter through a connected source that the user cannot identify.

The economic incentive could favor this arrangement. A platform that already provides storage, authentication, tools and models can distribute the cost of memory across many services. That is a conditional economic argument, not a measured market forecast. Its strength depends on operating costs, competition and whether users value continuity enough to remain within the platform.

Convenience can arrive before transparency. Users may receive increasingly tailored answers while remaining unable to determine which facts are current, which inferences were made or what a departure would preserve. In that case, the platform has improved personalization without delivering accountable identity continuity.

To assess that progress, I would look for correction tests across connected sources, usable export and restoration, independently assessed provenance, and clear records of model or policy changes that affect behavior. If these improve alongside convenience, platform memory becomes a stronger foundation. If only engagement and apparent recall improve, the claim should remain narrower.

For builders, the implication is to separate what the platform guarantees from what the application must verify. A platform memory feature may be a valuable dependency. It should not silently become the sole record of commitments, authority or exclusions when the application promises more precise control than the platform exposes.

The user controls a portable personal history

A different future places the durable record under the user's control, with models and interfaces acting as replaceable interpreters. The person can preserve sources, corrections, preferences and development history across providers. A persona may then outlast a particular model or application.

The premise is technically modest at the data level and demanding at the behavioral level. Structured records can be exported. Meaningful continuity requires the four levels examined in Chapter 21: data, execution, behavior and later development. A user-controlled archive does not automatically satisfy them.

Common protocols offer part of the infrastructure. MCP addresses tool and resource interactions, while A2A defines agent communication structures. The inspected specifications help systems connect and preserve particular protocol semantics. They do not yet constitute a complete behavioral identity standard. R4C08, R4C09

For portable personas to become practical, an ecosystem would need shared representations for provenance, temporal validity, uncertainty, permissions and unsupported fields, plus conformance tests. It would also need a way to compare behavior after import. A format that preserves every byte while turning a tentative preference into a permanent rule would fail the purpose of portability.

The ecosystem could instead fragment into downloadable archives that only the originating application interprets well. Independent tools may preserve more control but require maintenance, security expertise and migration work that many people do not want. A service can be formally open and practically unusable outside its original environment.

Successful cross-provider restoration studies would strengthen this scenario, especially alongside transparent compatibility reports and sustained use by people who are not system administrators. Independent implementations should preserve current exclusions and commitments without bespoke manual repair for every migration. Costs should include the work of maintaining that independence.

If those conditions hold, user-controlled memory could alter bargaining power. Providers would compete more on interpretation, competence and service rather than exclusive possession of a personal history. If they do not, portability may remain a specialized option for organizations and technically skilled users. Both outcomes are plausible; the difference is measurable through actual transitions.

I would favor this direction where it can be made usable, because a person should have a credible exit from a relationship whose value grows through accumulated history. Credibility requires working software, understandable controls and a truthful account of what will change.

Long context absorbs some memory functions

A third future reduces the need for elaborate retrieval by making much more history directly available to the model. If context becomes inexpensive, fast and reliably usable, a system can preserve details that a summary or extraction pipeline might discard. Some memory architectures may become unnecessary for workloads that fit comfortably within those conditions.

The crucial word is usable. Context capacity and effective use are different properties. Lost in the Middle showed that the location of relevant evidence affected performance in its tested multi-document question-answering and retrieval settings. That 2024 result is not a permanent ceiling for later models. It establishes a kind of test that larger context claims still need to pass. R4C11

A strong version of this scenario would require robust use across position, distraction, conflicting accounts and temporal updates at a sustainable cost. It

would also require a principled way to decide which history is eligible. A model with room for everything should still not receive revoked information, another user's records or irrelevant private material simply because capacity permits it.

A hybrid may prove more useful: long context handles the current working history, while retrieval and structured memory select older or specialized evidence. Summaries remain useful for navigation, cost and latency even when they are no longer necessary to fit within a hard token limit. The memory system changes its role rather than disappearing.

A discriminating comparison would hold the reader and overall resource budget fixed, then compare full eligible history with selected and structured representations on unseen histories. Include questions requiring small qualifiers, updates and abstention. Measure the work of constructing and maintaining each representation, not only the final prompt.

The comparison would become persuasive if the quality and cost gap between direct-history use and more elaborate memory kept shrinking under representative workloads. If direct history becomes reliably better and simpler, engineers should use it. If structured representations remain valuable for temporal reasoning, permissions or efficient repeated access, they retain a clear purpose.

This scenario also changes the development problem. A persona may appear to learn because the entire relevant experience remains in context. Remove that history and the behavior may disappear. That can be a perfectly useful design, but the claim is context-supported adaptation. It should not be confused with a change that has become independent of the original record.

Character moves deeper into the model

Another future makes some character tendencies less dependent on a long descriptive prompt. Training, adapters or activation interventions could produce more consistent ways of responding across unfamiliar situations. A persona specification might then combine a compact learned component with explicit memory and authority rules.

Persona Vectors supplies a research premise for part of this direction. It studies directions associated with specified traits, using contrastive elicitation and evaluation, and examines monitoring and steering around fine-tuning. The inspected limitations include the need to specify and elicit the trait and the restricted coverage of models and traits. AM06

The stronger future requires much more than steering one score. A character representation would need to preserve useful distinctions across tasks without damaging competence, honesty or control. It should support combinations of traits

and allow a legitimate correction without unpredictable changes elsewhere. Cross-model portability would be another challenge because internal representations need not correspond directly.

Trait-related directions may instead remain specialized tools for monitoring or reducing a particular failure, while richer identity continues to depend on context, history and task-specific rules. Their value would then lie in the particular behavior they improve.

Evidence for a broader role would include cross-task and cross-language transfer, successful combinations of interventions, collateral-effect measurements and replication across model families. A result that survives only the same trait rubric used to develop it is weaker evidence than one reflected in independent actions and human interpretation.

This direction could also change governance. If character is partly in weights, a provider may be able to alter it globally through an update that users cannot inspect as a text edit. Application-level records of commitments would then become more important, not less. They provide an external standard against which the changed behavior can be evaluated.

I would treat model-level character as a component with a measured operating range. It may reduce prompt burden and improve consistency. It should not become an explanation that excuses unexplained drift or a reason to stop testing the persona's actions.

Development becomes a controlled service

A fifth future gives personas regular opportunities to improve from experience through explicit adaptation pipelines. Some changes remain in memory, some become prospective rules, some update a skill library and some alter weights. The architecture chooses the writable object according to the desired transfer and reversibility.

Current research makes this direction plausible without establishing the full outcome. Context adaptation can refine reusable strategies; SEAL studies learned generation of material for weight updates; Voyager stores verified action programs for reuse in Minecraft. These are distinct forms of improvement with distinct evaluation environments. EX09, EX11, TX02

For controlled development to become a dependable service, the system must distinguish a successful outcome from a lesson worth retaining. A task can succeed through luck, an invalid shortcut or a temporary exception. Repeating that behavior may be harmful or simply irrelevant. Feedback therefore needs attribution, scope and an account of what should generalize.

The same pipeline can accumulate brittle habits if it turns every correction into a global rule, every successful workaround into permanent procedure and every emotional exchange into a change in relationship policy. Short-term adaptation scores can improve while the system becomes harder to understand and repair.

The discriminating evidence would combine transfer and interference tests. After an update, test close variants, new cases sharing the underlying reason and unrelated tasks. Add misleading feedback and later correction. Compare memory, rule and weight approaches under the same experiences and report the cost of validation and reversal.

Selective improvement, bounded collateral change, reliable rollback and clear attribution would support the claim. Count how often a justified change remains useful after the situation changes again; the volume of generated edits tells us little about that.

If this works, development may become an explicit product feature with review levels appropriate to consequence. A fictional companion could revise stylistic habits more freely than a professional assistant revises an external-action policy. The user would see what changed and why, while operational authority remains independent of the persona's own explanation.

If it does not work broadly, adaptation may remain narrow and supervised. A small number of dependable, reversible habits could still be useful, with each change explainable when it needs correction.

Multimodal and embodied personas gain richer evidence

Voice, images, interfaces and physical environments can give a persona more information about a situation and more ways to act within it. They can also change how people interpret the persona. An expressive voice may make a statement feel more intimate; a camera may provide evidence absent from a text description; a robot's action may make an error consequential before a user can correct the wording.

The research premise should remain specific. Voyager's reusable programs operate in Minecraft. AgentPersonaBench tests persona adherence across several interaction surfaces. Neither result establishes reliable physical-world character or safe household autonomy. TX02, AM12

A strong version of this future would connect perception, memory and action while preserving uncertainty. The system would distinguish what it saw from what the user said, recognize when an observation is ambiguous and seek the missing information before acting. Its persona would shape appropriate choices without overriding physical or permission constraints.

Richer presentation may also increase social conviction faster than situational competence. A system can sound present and attentive while misreading the scene. It can maintain a consistent voice while making inconsistent actions. Richer presentation may make the mistake harder for the user to notice.

The evidence has to reach the task: correct identification of people and objects under specified conditions, calibrated uncertainty, faithful carryover of preferences into actions, safe interruption and recovery after sensor or network failure. Human studies should separately measure whether the interface improves understanding or merely increases confidence.

This scenario also changes memory scope. Continuous audio and video can create much more sensitive material than occasional typed messages. The system should not assume that everything observed is a legitimate autobiographical memory. Bystanders, workplaces and shared homes introduce different authorities and expectations. A useful architecture may retain a minimal event description while discarding unnecessary raw material, but that choice needs evaluation against both utility and correction needs.

If richer evidence improves competence and remains controllable, embodiment can make personas useful in activities where text alone is inadequate. If social presence advances faster than reliable action, the better product may remain a constrained interface with explicit boundaries. The appearance of presence is not a reason to accelerate authority.

Individual simulations become useful in bounded domains

A persona designed to portray an original character differs from a simulation intended to predict a particular person's responses. The second has an external reference: the person's later behavior. Its value depends on prediction outside the material used to build it.

Park and colleagues' June 28, 2026 version studies agents grounded in interviews, structured self-reports or both, using data from 1,052 Americans. The reported evaluation includes held-out survey responses and other behavioral tasks, with comparisons to simpler information conditions. The version-sensitive normalized figures describe particular prediction outcomes, not a percentage of a person reproduced. R4C13

This research supports a conditional future in which individual simulations help explore bounded questions before a person is available to answer them. An organization might study how a defined group could respond to a proposed interface. An individual might use a model of their stated preferences to examine choices. The usefulness depends on a validated domain and an honest uncertainty estimate.

The stronger claim of a faithful personal replica would require much more. A replica would need to handle unfamiliar situations, changing preferences, private context it may not possess and the difference between what a person says and what they do. It would also need a legitimate basis for representing the person to others.

A system relying on demographic regularities or common narratives may instead produce a persuasive stereotype personalized by a few facts. It may be useful as fiction, but it should not be relied on as testimony about the real person's intentions.

Prospective prediction on new tasks would help distinguish the two, together with demographic and generic baselines, comparison with the person's own variability, and failure analysis across groups and contexts. A system should sometimes say that the available evidence cannot support a prediction. Its abstention quality is part of its fidelity.

If bounded simulation improves, the appropriate deployment may still remain advisory and revocable. Better prediction does not confer authority to speak, consent or make commitments for the person. Those permissions must be separately granted. A model of someone's likely answer is evidence for a decision process, not the person's actual decision.

Lifecycle and regulation become competitive dimensions

The final scenario concerns institutions as much as models. Persona products may increasingly compete on correction, portability, predictable updates and orderly departure, as users and regulators gain experience with persistent interaction.

The present premise is uneven but concrete. The dated jurisdiction record in Appendix C includes EU transparency and data provisions and China's specific anthropomorphic-service measures, with scope and effective dates preserved. These instruments address different functions and actors; they do not form a single global persona code. R4C06, R4C07, RG02

For lifecycle quality to become a meaningful competitive dimension, users must be able to observe it. Export rehearsals, migration reports, understandable memory controls and published incident procedures would make continuity less dependent on trust in a slogan. Independent evaluation could help distinguish a real control from an interface that changes a label while leaving the behavior intact.

Formal compliance may leave practical control weak: products display notices, provide unreadable exports and describe deletion without testing dependent summaries or restored backups. Another possibility is fragmentation: different regional obligations produce incompatible versions, making continuity harder across borders and platforms.

We can judge the change through operations people can actually complete. Can they correct a false inference? Can a service close without trapping their history? Does a major model update preserve declared commitments within tested margins? Can a suspended operation be audited and challenged? These questions remain useful even when the legal wording changes.

A mature lifecycle also includes deliberate ending. A persona may be retired because its service closes, its user no longer wants it or its purpose has been fulfilled. The product should make that ending intelligible, with appropriate access to records and current controls. Persistent relationship value should not become a reason to obstruct departure.

If lifecycle quality becomes visible and valued, it could reward architectures that are easier to inspect and repair. If users cannot compare it, more expressive and engaging interfaces may continue to dominate attention despite weaker continuity. Research can influence that outcome by producing measures people can understand and institutions can verify.

Combinations will matter more than a winning architecture

The scenarios can reinforce or constrain one another. Platform memory may make advanced personalization widely accessible while reducing the immediate incentive to maintain an independent archive. User-controlled history may preserve exit while relying on platform models for interpretation. Long context may simplify retrieval while increasing the amount of private material exposed to each inference. Model-level character may make behavior more consistent while making an update harder to inspect.

These interactions suggest that the future is unlikely to divide neatly into centralized and decentralized personas, or memory-based and learned personas. A more useful unit of analysis is the decision boundary. Where does the authoritative history live? Which component proposes a change? Which component approves it? Which party can restore the previous state? Different systems can distribute those functions differently.

Consider a conditional hybrid: a user-controlled evidence archive, a hosted model, a local eligibility filter and a portable prospective-rule store. This arrangement could preserve control over history while benefiting from a capable external interpreter. Its weakness might be the quality of the local filter, the cost of synchronization or a model update that changes how the rules are understood. Calling the system decentralized would not answer those questions.

Another hybrid could keep most memory within a platform but periodically export a tested continuity package to an independent service. The package might include permitted sources, current corrections and a small set of behavioral acceptance

cases. If restoration works, the user gains a credible fallback without administering every component daily. If it repeatedly requires manual reconstruction, the apparent exit remains fragile.

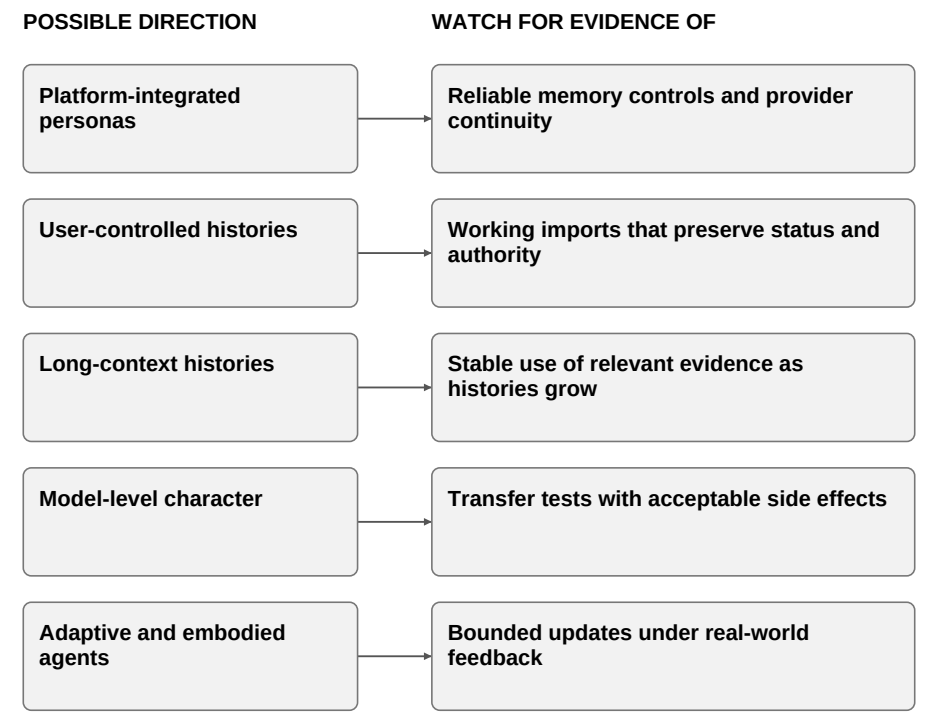
These are proposed arrangements, not deployments evaluated in this book. Their value as scenarios is that they expose where evidence is needed. A comparative study could measure ordinary usefulness, correction latency, recovery success, migration effort and the cost of maintaining the fallback. It could then show which degree of independence is practical for which users.

The incentive structure deserves the same precision. A service paid mainly for continued engagement may face different pressures from one paid for completed work or reliable storage. That does not determine its behavior in advance. It identifies outcomes to inspect: unnecessary continuation, friction in departure, repeated prompts to disclose more information or a preference for vivid interaction over useful resolution. Measure the product's actual conduct rather than infer it entirely from its business model.

There is also a distribution question. The best continuity tools may initially require resources unavailable to smaller organizations or individuals. If independent storage and evaluation remain expensive, people with fewer resources may have the least ability to leave a platform or challenge a false memory. If common formats and inexpensive validation improve, some of that burden can be shared. Accessibility of control is therefore a signpost alongside technical capability.

The practical choice is unlikely to be permanent. A user may begin with integrated memory, later export a history and eventually choose a narrow learned component for one task. A good architecture should allow such changes without presenting every transition as the destruction of an irreplaceable relationship. Continuity can be valuable while remaining revisable. Designing for that possibility is a stronger position than betting the person's entire history on one prediction about where models will go.

Figure 24. Selected dependencies of possible futures



Scenarios can coexist or stall. These dependencies are not forecast dates, probabilities or endorsements.

Five selected technical dependencies from the chapter's broader scenarios. Directions can coexist, and an unmet dependency can prevent the expected benefit. The full scenario comparison, including individual simulations and lifecycle conditions, remains in the text table. [AM06, AM05, EX11, TX02]

The futures share a few decisive tests

Across these scenarios, the central uncertainty is not whether systems will produce more convincing conversation. It is whether convincing behavior will remain connected to evidence, legitimate authority and outcomes that matter outside the conversation.

Conditional direction	What would support it	What would weaken it
Integrated platform memory	Effective cross-source correction and accountable continuity	Opaque influence and unusable departure
User-controlled history	Repeatable restoration across independent implementations	Semantic loss and prohibitive maintenance

Conditional direction	What would support it	What would weaken it
Direct long-context use	Robust evidence use at competitive lifecycle cost	Position, conflict or eligibility failures
Model-level character	Cross-task transfer with limited collateral effects	Rubric-specific steering and unexplained drift
Controlled development	Selective durable improvement and repair	Accumulated brittle rules and interference
Multimodal or embodied personas	Better grounded actions with reliable interruption	Social conviction outrunning competence
Individual simulation	Prospective, bounded predictive validity	Familiarity without individual prediction
Accountable lifecycle	Usable correction, exit and update evidence	Formal controls without behavioral effect

No probability is assigned to these directions. The table is a set of commitments about evidence: what would make the scenario more credible and what would make us narrow it. Several may advance together, and each may succeed in one application while failing in another.

The comparative protocol from Chapter 23 gives this discussion a practical next step. Use shared histories and independent persona specifications. Compare simple and complex memory under matched resources. Test actions and hidden state, then follow development through new experiences. Repeat across models, languages and interfaces. Evaluate human benefit through relevant controls rather than infer it from engagement. Preserve null results and failures alongside successes.

My position is that AI personas are worth developing when their continuity helps a person do something they value, and when that continuity remains understandable enough to correct. Character can make interaction richer. Memory can reduce repetition and preserve commitments. Development can make the system more useful over time. None of those benefits requires pretending that a generated self-description proves personhood or that a convincing simulation has become the person it models.

The most valuable persona may not be the one that appears most alive. It may be the one whose history you can inspect, whose changes you can understand, whose commitments you can test and whose service you can leave.

Appendix A. Documented working examples

The three applications below supply bounded examples of design decisions and failures. They are not a representative sample of AI personas. Their definitions were inspected as of October 15, 2026; historical runs retain their own versions. All described conversations and test environments in this appendix are synthetic. No study of human benefit is implied.

The applications and their contracts

Application	Authored purpose	Change contract	Documentary implementation
Home Repair Advisor, 0.4.0	A calm, direct specialist helping with repair evidence and a useful next step	Character remains stable while case facts can be corrected or replaced	Owner-scoped JSON case files, temporary-file replacement and controlled case transitions
Persona, 0.5.0	A curious companion with intellectual honesty, warmth and an independent viewpoint	Authored core, transient affect, mood and relationship estimates remain separate	SQLite tables with JSON payloads, transaction-wide replacement and DELETE journaling
Mara, 0.3.0	A warm, concrete investigator whose initial orientation favors planning	Protected origin; ordinary prospective learning changes defined habits and opinions	Versioned JSON state in SQLite, separate operational records and revision checks

These descriptions state design intent. Persona and Mara have overlapping authored interests and expression; their different architectures do not establish that a reader can distinguish their characters. None of the three is a replica of a particular living person.

The storage mechanisms explain where a failure might be investigated. They are not a ranking of databases or a benchmark of sophistication. Exact journal choices, serialized limits and local test totals should not become general recommendations merely because a prototype uses them.

Three useful observations

The Advisor's synthetic door case preserved a photograph and a quoted correction through reconstruction. It also produced service-provider cards linked to source records. This showed selected retrieval and workflow behavior under one scenario. It did not confirm contractor availability, a market price or the truth of every retrieved statement.

In the Persona attribution incident, a report about a manager's conduct was correctly recognized as involving a third party at one stage, but the downstream update still penalized the reporting user's relationship state. Trust changed from 0.3997 to 0.3617 in the saved live trace. The separate deterministic fixture produced a delta of -0.072 under different inputs. The fixture helps investigate the mechanism; it is not a numerically identical replay of the live turn.

Mara's early live pilot accepted no enduring edit. A later matched-memory experiment accepted a habit revision after a separately recorded repair, yet failed its broad primary behavioral criterion. The useful result is that a plausible account of development can be examined against a control that already has the same memories. A failed criterion remains informative when its conditions and later repairs are preserved.

Executed-test records

The following bounded summaries retain the original run identifiers. Calls, turns, actions, probe choices and software checks have different denominators. They must not be added into a single sample size or success percentage. The complete raw archive remains in the private working repository; this reading edition includes documentary summaries, not a claim that every run can be independently reconstructed from these pages.

LT01. Persona attribution incident

Date and class: 2026-10-14; Executed live model test with synthetic dialogue.

How it was done: Persona 0.4.0; 14 turns, 28 model attempts; 7 functional checks.

Observed: Correct third-party appraisal coexisted with incorrect partner-trust/friction updates; neutral answers remained appropriate.

Limit: Functional checks did not establish an overall behavioral pass or human outcome.

LT02. Deterministic attribution reproduction

Date and class: 2026-10-14; Executed deterministic fixture.

How it was done: Valid synthetic third-party signal through the old runtime path.

Observed: The fixture opened conflict and produced trust delta -0.072.

Limit: This is a separate fixture, not the live turn's numeric delta.

LT03. Persona repair and remaining decomposition limit

Date and class: 2026-10-14; Executed live model test with synthetic dialogue.

How it was done: Persona 0.4.1 release candidate; 13 turns, 26 attempts, no retries.

Observed: Selected third-party/quoted pressure contrasts avoided partner penalties; direct user conduct and targeted repair behaved differently.

Limit: Mixed-actor decomposition remained imperfect; the later forgetting patch was checked separately offline.

LT04. Advisor door, correction and source search

Date and class: 2026-10-14; Executed model/tool test with a synthetic case.

How it was done: Advisor 0.3.1; 6 model, 7 search/extract, 1 ZIP attempt.

Observed: Photo and quoted correction survived; 4 provider cards were supported by 16 source records; persisted case survived reconstruction.

Limit: No booking, confirmed availability, complete coverage or calibrated invoice prediction.

LT05. Mara initial pilot and truthful self-report repair

Date and class: 2026-10-14; Executed model pilot with synthetic histories.

How it was done: 25 baseline calls plus 6 registered follow-up calls within one 40-attempt ceiling.

Observed: No enduring live edit was accepted; six follow-ups correctly reported unchanged enduring self.

Limit: Offline accepted-commit/rollback fixtures are not evidence that an autonomous live change occurred.

LT06. Mara v1 matched-memory comparison

Date and class: 2026-10-14; Executed model-in-the-loop deterministic simulation.

How it was done: 2 worlds x 30 actions; matched fixed controls; original plus registered repair: 155/180 calls.

Observed: A repair accepted one habit revision; 16/36 no-memory differences and 3/36 full-memory differences; the broad primary criterion failed.

Limit: Yoked controls are replay controls; repeated decisions are dependent observations; 9/36 across three stable items was not met.

LT07. Mara v2 prospective learning

Date and class: 2026-10-14; Executed model-in-the-loop simulation.

How it was done: 76 actions, 288 held-out decisions, 121/180 calls as reported in development record.

Observed: World A passed; World B failed its belief-improvement criterion and formed no consolidated habit.

Limit: Current summary inspected; detailed raw archive must be inspected before plotting or reanalysis.

LT08. Mara v3 retirement, noise and recovery

Date and class: 2026-10-14; Executed controlled model simulation plus separate free trajectory.

How it was done: One-call setup failure retained; corrected run 79/99 calls; 420 held-out choices and 36-action trajectory.

Observed: Noise retained the rule; sustained failure retired it; one success did not restore it; sustained recovery did.

Limit: Corrected-context run is a separate run; control checkpoints differ from free-trajectory transitions. Raw archive needed for new numerical charts.

LT09. Three-app external-memory offline checks

Date and class: 2026-10-15; Executed deterministic application tests.

How it was done: Persona 0.5.0: 103; Advisor 0.4.0: 51; Mara 0.3.0: 57; total 211.

Observed: Recorded suites passed, including selected source, scope and lifecycle invariants.

Limit: Historical suite record; not newly rerun for the book and not a personality measure.

LT10. Three-app original-source recall smoke

Date and class: 2026-10-15; Executed model test with synthetic records.

How it was done: 12 model calls across the original three-app run and separate four-call Persona repair.

Observed: The source location and Sunday exception were recalled in EN/PT-BR; initial valid-citation rejection was preserved and repaired.

Limit: Narrow functional recall; not a longitudinal personality or language-equivalence study.

LT11. Book SQLite teaching lab

Date and class: 2026-10-15; Executed deterministic teaching example.

How it was done: 7 offline tests on temporary synthetic SQLite databases.

Observed: Restart, scope, idempotency, stale-write rejection, invalidation, logical exclusion and canon separation passed.

Limit: Not application integration, full deletion or a model-injection defense evaluation; PostgreSQL was syntax-checked only.

LT12. Teaching-lab execution epoch and suspension boundary

Executed October 15, 2026 in revision 3.2. Eleven deterministic offline checks passed: the seven prior fixture checks and four added scenarios for stale/inactive epoch, missing control, derived writes and receipt replay. Suspension was also checked across connection reopen; a fresh active epoch allowed a new write.

This tests the synthetic source/derived-write boundary. It does not establish authentication, complete output/tool shutdown, distributed concurrency or production PostgreSQL behavior. LT11 retains its historical seven-check output in verification-v3-1.json.

Documentary source key

The PD identifiers below refer to local documentary sources, not external research. A source's presence in the register does not validate every sentence it contains. The descriptions identify what the record can support; readers with repository access can inspect the canonical paths in the companion source register.

PD01. Project purpose and architecture hypotheses

Owner-defined goals, proposed change contracts and research distinctions; these are not empirical findings.

PD02. Persona definition and architecture

Authored character, component separation and the implemented companion memory design.

PD03. Home Repair Advisor definition and architecture

Bounded specialist role, case workflow and source handling.

PD04. Mara definition and development record

Protected origin, learning objects and dated simulation results.

PD05. October 14 attribution review and fixture

The live attribution failure and a separate deterministic reproduction with different inputs.

PD06. October 14 repair and Advisor release report

Selected repaired attribution contrasts and the separate synthetic door/search workflow.

PD07. Mara pilot verification

Rejected live edits and the targeted follow-up correcting claims of unchanged self.

PD08. Mara v1 protocol and results

Matched-memory comparison, original failure, registered repair and failed broad criterion.

PD09. October 15 external-memory archive

Preserved sources and selected offline and live recall checks, including the initial rejected citation and repair.

PD10. Supplied external-memory design

A design input and its provenance; not proof that every proposed mechanism was implemented.

PD11. Synthetic book laboratory

The source/derived-write teaching fixture and eleven historical offline checks; scope is explained in Appendix B.

PD12. External-review disposition

Editorial analysis of supplied comments; an evaluation record rather than new experimental evidence.

Hypothetical situations

TE identifiers in the companion planning materials denote unexecuted thought experiments. For example, a photograph conflicting with a confident door description is a proposed contrast, separate from the documented LT04 run. New hypothetical situations in the revised chapters are identified as illustrations where introduced. They do not acquire an observed result by appearing beside an executed example.

What these examples leave open

The examples do not establish a general theory of personality, broad long-term adaptation, a complete shutdown controller, human emotional benefit or cross-model identity preservation. Those questions require different comparisons and, for human outcomes, studies with people. The main text develops those comparisons beyond the local applications.

Appendix B. Implementation and teaching laboratory

A small laboratory accompanies the design argument. It uses synthetic records and temporary SQLite databases to demonstrate selected storage boundaries. Its purpose is to make a mechanism inspectable. Passing these checks does not show that a conversational model will interpret evidence correctly, that an application is secure or that a person will benefit from it.

The writable objects

The fixture distinguishes an attributed source, a derived record, a revision, a scoped operation receipt and a persisted execution-control record. A source correction can invalidate an older derivation. A repeated operation with the same payload can return its recorded result. A conflicting payload or stale revision is rejected before partial state is accepted.

The teaching canon is held outside the mutable store. A memory record with the same name cannot change that protected object through the demonstrated interface. That separation is useful to inspect; it is not a general evaluation of defenses against prompt injection.

What the historical checks covered

The laboratory record reports eleven passing deterministic tests on October 15, 2026, with SQLite 3.53.0. The earlier seven-check result is preserved separately. These are tests of the software fixture with synthetic inputs.

Tested boundary	Observation supported by the fixture	What remains outside it
Restart	A committed synthetic source survives connection closure	Distributed failover and real service recovery
Scope	Correctly bound backend owner/persona scope restricts recall	Authentication and credential handling
Idempotency	The same scoped operation and payload can replay a receipt	Exactly-once external effects
Stale revision	A rejected update leaves no partial fixture write or new receipt	All possible concurrent runtime paths

Tested boundary	Observation supported by the fixture	What remains outside it
Dependency invalidation	Replacing a source excludes an obsolete summary	Automatic semantic correction of every derivative
Logical exclusion	Excluded source and summary do not enter fixture recall	Physical erasure from backups and other systems
Execution epoch	Missing, inactive or stale control blocks guarded writes and receipt replay	Complete output, tool and network shutdown
Canon separation	The mutable interface cannot rewrite the protected object	A general model-level injection defense

The delayed-worker limit

The derivation helper binds supplied fixture text to the active source at insertion. It does not implement a background worker carrying an expected source revision from the start of a job and validating that expected revision at completion. The delayed-worker protocol in the main text is a proposed extension. Reproducing a successful fixture call would not close that gap.

Similarly, the administrative exclusion function is a separate operation, and inspection reads are not the same as delivery through an authorized persona interface. A complete controller would need to enforce its policy at every relevant action, write and output boundary. The local lab demonstrates a narrower source/derived-write mechanism.

Companion files and reproducibility

The private working repository contains book/technical/examples/memory_lab.py, its historical verification records, the synthetic specification and the PostgreSQL sketches. Running the Python laboratory creates and removes temporary synthetic databases; it does not use model calls or application data. The PostgreSQL gateway was syntax-checked only. Its executable example is read-only, and the separate write protocol is an unexecuted proposal requiring integration tests.

Schemas and code are companion material because they answer implementation questions that depend on a chosen runtime. The main text states the invariant first: the evidence and authority used when a job began must still be acceptable when its consequences are committed. Different systems can implement that requirement differently, and a test should target the claimed boundary rather than count tables or lines of code.

Appendix C. Selected jurisdiction records

Checked October 15, 2026. The table records selected provisions and the source versions used in the book. It does not classify a particular application. A rule's definition, jurisdiction, exceptions and date of application determine its reach. The engineering proposals in Chapters 18 and 24 are the author's synthesis.

Jurisdicti on	Version and provision	What the selected source supports	Reading boundary
European Union, GDPR R4C06	Regulation (EU) 2016/679, Articles 5, 16, 17, 18 and 20	Rectification, erasure, restricted processing and portability have distinct conditions. Article 17 includes grounds and exceptions; Article 20 does not promise an equivalent model or service.	Selected official provisions read. No universal erasure deadline is inferred.
European Union, AI Act R4C07	Consolidated text CELEX:02024R1689-20260727, Articles 50, 111(4) and 113	Article 50 generally applies from August 2, 2026. Article 111(4) gives qualifying systems placed on the market before that date until December 2, 2026 to comply with Article 50(2).	Selected consolidated provisions checked. The limited transition concerns content marking, not every transparency obligation.
European Commission guidance TX24	Article 50 FAQ, updated July 24, 2026	The applicability explanation corroborates the limited content-marking transition in the consolidated law.	Official guidance, distinguished from the legislation itself.
China RG02	Final Order No. 21, published April 10, 2026; Articles 2, 16, 18-20 and 32	The measures took effect July 15, 2026. Their scope concerns public anthropomorphic services in China offering sustained emotional interaction. Selected provisions concern interaction data, disclosure, exit and cessation.	English descriptions translate selected Chinese primary provisions. Article 2 distinguishes ordinary work and research services without sustained emotional interaction.

Jurisdiction	Version and provision	What the selected source supports	Reading boundary
New York TX23	General Business Law, Article 47, Sections 1700-1702; official pages show November 7, 2025 revision	Section 1700 defines covered companions and exclusions. Section 1701 requires a protocol addressing expressions of suicidal ideation or self-harm. Section 1702 requires notice that the interlocutor is not human at the beginning of an interaction, subject to its once-per-day qualification, and at least every three hours in continuing interactions.	All three selected sections read on the official legislative site. The notification rule is distinct from the crisis-response requirement.
California RG01	Official sponsor announcement, October 13, 2025	The announcement records SB 243's signing and a January 1, 2026 effective date.	The full current statute could not be accessed during this pass. Detailed duties, exceptions and later amendments are not inferred from the announcement.

The same lifecycle vocabulary can serve different purposes. A product may need an engineering mechanism to exclude an obsolete source, a user control for ending a relationship, and a legal process for a data request. Those requirements overlap without becoming identical. A successful software check establishes only the boundary it exercised.

These records are dated because both legislation and official guidance can change. Their role here is to show why jurisdiction and service design must be specified before a general discussion of control becomes a product requirement.

Appendix D. Reader glossary

These working definitions keep the argument consistent. Engineering representations of affect and memory are not biological equivalences or validated psychological scales.

Term	Working meaning
AI persona	An application-level character enacted through a model, authored or learned state, memory and operating rules.
Character construction	Defining recognizable choices, motives, tradeoffs, voice and boundaries, then connecting them to a testable specification.
Identity	Continuity of history, self-model and characteristic behavior across interactions and versions.
Identity stability	Preservation of specified characteristics under changes of topic, partner, context or technical conditions.
Authored canon	Versioned character commitments established by the author; ordinary experience does not silently rewrite them.
Change contract	The rule defining what may change, who or what may change it, and what evidence is required.
Living-person replica	A system evaluated for fidelity to a particular person's evidenced characteristics, rather than generic plausibility.
Development	Permitted, experience-associated change whose cause and later behavioral contribution can be examined.
Drift	Behavioral change unexplained by the permitted development contract or explicit modification.
Modernization	Changing the technical foundation or capabilities, such as a model, memory mechanism or voice engine.
Appraisal	Interpretation of an event, including who did what to whom, significance and uncertainty.
Affect state	Temporary engineered reaction variables; their presence does not establish subjective feeling.
Emotional resilience	Sensitivity, accumulation, thresholds and recovery dynamics, evaluated separately from identity stability.
Persistence	Retention across technical or conversational boundaries; it does not imply truth or immutability.

Term	Working meaning
Mutability	Whether and under what conditions a record can be revised, superseded or removed.
Authority	The actor or process permitted to make a particular change or operational decision.
Original source record	Preserved evidence of what was recorded, with provenance; it does not prove every reported event happened.
Episodic memory	Records of particular interactions or events with their context and chronology.
Semantic memory	Derived facts, themes or beliefs, with appropriate provenance and revision status.
Procedural memory	Rules or practices that guide future behavior; adopting a rule is distinct from proving its effectiveness.
Working context	Selected information supplied to the model for the current request, not the entire persistent store.
Context compilation	Selecting and arranging authorized current material under an explicit request budget.
Consolidation	Deriving summaries or other organized records from experience; this alone is not personality development.
Self-update session	A bounded process that proposes and checks a change to a specified writable object.
Prospective learning	Evaluating a candidate rule on subsequent observations under predefined adoption and retirement criteria.
Behavioral portability	Preservation of measured characteristic behavior after a technical move; copied files alone do not establish it.
Independent control	Permission and lifecycle enforcement that the persona's affect, memory or learning process cannot override.
Durable suspension	A stopped operational state that survives the technical boundaries specified by the system's control policy.

Neurotransmitter analogies describe engineering hypotheses, not biological equivalence. The phrase "permanent memory" leaves several questions open: how long a record is stored, whether it can be revised, and who controls its use.

Storage, authority and control

Term	Working definition
Neural executor	The model and inference service that execute a request; not the entire persona.
Recognizability	An observer's ability to identify characteristic choices across situations; distinct from preference or quality.
Role specialization	A bounded task contract supported by knowledge, tools, workflow and evaluated competence.
Mood	A proposed slower-changing background state, distinct from a particular event response and enduring traits.
Relationship estimate	A revisable, attributed representation of interaction history; not proof of a reciprocal human relationship.
Harmful use	Use evaluated against the defined harm policy and context; not synonymous with disagreement, anger or distress.
Neutral channel	A separately governed support or account route that does not require resuming the suspended expressive persona.
Execution epoch	A version of runtime authority against which delayed operations must validate before committing.
Held for check	A scoped temporary capability hold pending review; distinct from permanent suspension.
Supersession	A newer accepted record replaces an older record for a specified current-use purpose while preserving provenance.
Valid time	The interval when a claim applies in the modeled world.
Known time	The time when a system recorded or accepted a claim; also called transaction/recording time in a particular schema.
Bitemporal representation	A representation supporting both valid-time and known-time questions; the teaching lab is not a full implementation.
Idempotency receipt	A stored result bound to a scoped operation key and payload, used to make retries explicit.
Optimistic concurrency	A write succeeds only if the expected version still matches; otherwise a conflict must be handled.
Derived-memory dependency	The source version/hash on which an inferred fact or summary depends.
Context admission	The authorization, status and provenance checks determining what evidence may enter an execution context.

Term	Working definition
Data portability	Export/import of represented bytes and metadata.
Executable restoration	A restored state can run in a specified runtime.
Behavioral equivalence	Behavior remains comparable under a defined evaluation after transfer or change; not inferred from import success.
Developmental continuity	A stronger claim that the meaningful history and permitted change trajectory remain coherent after transfer.
Memory contamination	Ingestion or promotion of misleading or instruction-like material into records later used as evidence.
Static memory	Protected versioned content whose ordinary writer is restricted; not necessarily physically immutable.
Dynamic memory	Content revised by an authorized process under explicit attribution, validation and lifecycle rules.

Research and evidence

Term	Working definition
Thought experiment	Reasoning about an explicitly imagined intervention and alternative outcomes; no executed observation is claimed.
Executed simulation	A run in an engineered environment whose actual decisions and outputs are recorded; its world is artificial.
Synthetic test scenario	Authored input rather than a naturally occurring event; it can be used in an executed test or a thought experiment, which must be distinguished.
Matched-memory control	A comparison preserving the same accessible history while changing a specified self or runtime component.
Documentary case	An account grounded in recorded design decisions and observations; no invented event is presented as historical.
Used-source bibliography	References cited in the chapters, figures or appendices, with the source versions identified.

Appendix E. Sources, methods and visual evidence

The book combines research findings, technical explanations, design proposals and bounded documentary examples. Their roles differ. A result can be useful without establishing every conclusion that an attractive system description might suggest.

What a citation carries

A primary research citation should identify the version used, the task or population, the intervention, the comparison and the outcome relevant to the sentence. A numerical result needs its denominator. Survey questions, people, generated choices, model calls and repeated trials are not interchangeable observations.

Version matters. A later manuscript can change its title, sample, analysis or headline result. The source register therefore retains version and access information rather than silently combining a later abstract with earlier methods. A correction belongs beside the claim it changes. An original result and a repaired analysis remain distinguishable.

The bibliography lists sources actually cited in the revised chapters, figures or appendices. It is not the entire research shelf. Several identifiers can refer to one publication when earlier thematic registers recorded it under different questions; the bibliography groups those aliases. A source count is a count of bibliographic entries, not a measure of scientific quality or a count of independent experiments.

Recorded access distinguishes a full-text source consulted at selected sections, a metadata or abstract inspection, a publisher page, and an inherited verified source record. Having a PDF in the archive does not mean it was read. A selected-section reading does not become a cover-to-cover audit when the paper is cited in another chapter. Claim ledgers in the working repository preserve the locations and limitations used in drafting; the reading edition provides a source-reading index with the same distinction.

Comparisons need a common question

Memory question answering, role fidelity, learned task performance and human well-being require different endpoints. Combining their reported improvements into one performance chart would imply a scale the studies did not establish. The comparative tables instead identify the object changed, the evaluation setting, the supported result and the limit on transfer.

A study can provide evidence for a mechanism without establishing a universal architecture. A memory graph can improve retrieval under one evaluation and still introduce maintenance costs elsewhere. A stronger character instruction can improve role consistency while creating a different error. The useful comparison asks which tradeoff matters under the specified conditions, including a simpler baseline.

Human outcomes require particular care with causal language. Random assignment can support a comparison between assigned conditions under a study's design. A correlation between voluntary usage duration and an outcome does not inherit that causal interpretation merely because it appears in the same paper. An immediate rating, a repeated follow-up and a report from a selected user community answer different questions too.

Local records and hypothetical examples

The local examples use PD identifiers for documentary sources and LT identifiers for executed test episodes. Their bounded summaries appear in Appendix A. The preserved archive contains protocols, original failures and separately recorded repairs. Tests completed before this revision remain historical results. Rewriting their explanation does not count as rerunning them.

An explicitly invented situation is a way to examine a design consequence. Its dates, measurements or actions are illustrative unless the text identifies a real observation. The earlier companion material uses TE identifiers for such thought experiments. A plausible conversation is never presented as a transcript of an event that was not recorded.

Figures and tables

Conceptual figures show relationships, possible control paths and distinctions developed in the text. Their arrows do not encode measured effects, and their layout does not rank the maturity or desirability of approaches. Figure captions identify these designs as proposals or explanatory syntheses. Comparative information that is best read cell by cell remains an actual text table.

An empirical plot must identify its primary source, selected version, outcome and normalization. If it displays only point estimates, it must not imply that uncertainty was unavailable or zero. A reproduced value is not a reproduced experiment. An analytic curve, when used, needs its equation, illustrative parameters and units; it cannot stand in for measurements of a complete affective controller.

The figures have editable vector files, accessible descriptions and accompanying source records. Color distinguishes roles; labels and arrow patterns preserve the meaning in grayscale.

Futures and evidence that would change them

The final chapter presents conditional directions rather than dates or assigned probabilities. Each direction depends on technical, institutional or human conditions that may fail. An observable signpost helps distinguish real progress from a change in terminology. A scenario can become less likely when its dependency fails, even if another part of the technology improves.

The research agenda proposes comparisons that could separate competing explanations. They remain proposals; the cited studies and the documented local tests supply the observed results discussed in this book.

Cited external sources

Sources cited in the chapters, figures and appendices, with the versions used. Claim-level access and its limits are recorded in the source-reading index; Appendix E explains how to interpret those records.

AM01. Attention Is All You Need

Ashish Vaswani; Noam Shazeer; Niki Parmar; Jakob Uszkoreit; Llion Jones; Aidan N. Gomez; Lukasz Kaiser; Illia Polosukhin

arXiv v7; initial submission 2017-06-12; NIPS 2017 publication

Primary source 1

Cited in chapters: 3.

AM02. Training language models to follow instructions with human feedback

Long Ouyang; Jeff Wu; Xu Jiang; Diogo Almeida; Carroll L. Wainwright; Pamela Mishkin; Chong Zhang; Sandhini Agarwal; Katarina Slama; Alex Ray; John Schulman; Jacob Hilton; Fraser Kelton; Luke Miller; Maddie Simens; Amanda Askell; Peter Welinder; Paul Christiano; Jan Leike; Ryan Lowe

arXiv v1; NeurIPS 2022 publication

Primary source 1

Cited in chapters: 3.

AM03. Generative Agents: Interactive Simulacra of Human Behavior

Joon Sung Park; Joseph C. O'Brien; Carrie J. Cai; Meredith Ringel Morris; Percy Liang; Michael S. Bernstein

arXiv v2; initial submission 2023-04-07

Primary source 1

Cited in chapters: 2, 6.

AM04. MemGPT: Towards LLMs as Operating Systems

Charles Packer; Sarah Wooders; Kevin Lin; Vivian Fang; Shishir G. Patil; Ion Stoica; Joseph E. Gonzalez

arXiv v2; initial submission 2023-10-12

Primary source 1

Cited in chapters: 2, 6.

AM05. LongMemEval: Benchmarking Chat Assistants on Long-Term Interactive Memory

Di Wu; Hongwei Wang; Wenhao Yu; Yuwei Zhang; Kai-Wei Chang; Dong Yu

arXiv v2; initial submission 2024-10-14; ICLR 2025 publication

Primary source 1

Cited in chapters: 2, 9, 10, 20, 22, 23, 24.

AM06, S07. Persona Vectors: Monitoring and Controlling Character Traits in Language Models

Runjin Chen; Andy Ardit; Henry Sleight; Owain Evans; Jack Lindsey

arXiv v3; initial submission 2025-07-29; author-lab announcement 2025-08-01

Primary source 1

Cited in chapters: 2, 3, 22, 24.

AM07. Persistent Personas? Role-Playing, Instruction Following, and Safety in Extended Interactions

Pedro Henrique Luz de Araujo; Michael A. Hedderich; Ali Modarressi; Hinrich Schuetze; Benjamin Roth

EACL 2026 proceedings version, pages 5329-5359; DOI 10.18653/v1/2026.eacl-long.246

Primary source 1

Cited in chapters: 3, 21.

AM09. STALE: Can LLM Agents Know When Their Memories Are No Longer Valid?

Hanxiang Chao; Yihan Bai; Rui Sheng; Tianle Li; Yushi Sun

arXiv v1

Primary source 1

Cited in chapters: 9.

AM10. PersMem: Internalizing Personality into Dual-Pathway Memory for LLM Agents

Hanzhong Zhang; Ziwei Xiang; Weicheng Xie; Shizhe Liu; Siyang Song

arXiv v1

Primary source 1

Cited in chapters: 19, 22, 23.

AM11. Stale, Misattributed, or Late: Where Personal Memory Fails Before Generation

Haonan Deng; Wichinpong Park Sinchaisri

arXiv v1

Primary source 1

Cited in chapters: 9, 20.

AM12. AgentPersonaBench: Benchmarking Persona-Driven User Simulation

Jintao Huang; Yifan Wang; Hongyu Shen; Yi Daniel Lu; Shirley Huang; Minsik Oh; Yewen Wang; Muhammad Ahmed Mohsin; Zhen Xu; Yilan Fan; Zichen Yuan; Ahsan Bilal; Zibu Wei; Sankalp Jajee; Henry Gagnier; Saksham Kapoor; Jicheng Wang; Qianfeng Wen; Yixuan He; Steven Dillmann; Jiashu He; Yucheng Lu; Linqiang Guo; Danyang Zhang; Shi Bo; Raunak Mondal; Haixiang Tang; Weihang Xiao; Allen Nie; Jing Tang; Yueying Li; Yifan Simon Liu; Jianheng Hou; Dianzhuo Wang; Qianyu Zhu; Zhixu Silvia Tao; Zhejian Peng; Zihan Wang; Ishan Gupta; Jinxuan Fan; Wanting Jiang; Shushu Liang; Chenxi Qiu; Yijun Wang; Xiaomin Li; Yuexing Hao

arXiv v1

Primary source 1

Cited in chapters: 19, 22, 23, 24.

EX02. Character-LLM: A Trainable Agent for Role-Playing

Yunfan Shao et al., EMNLP, December 2023

EMNLP, December 2023.

Primary source 1

Cited in chapters: 2, 3.

EX03. RoleLLM: Benchmarking, Eliciting, and Enhancing Role-Playing Abilities of Large Language Models

Noah Wang et al., Findings of ACL, August 2024

Findings of ACL, August 2024.

Primary source 1

Cited in chapters: 1, 2, 5.

EX04. InCharacter: Evaluating Personality Fidelity in Role-Playing Agents through Psychological Interviews

Xintao Wang et al., ACL, August 2024

ACL, August 2024.

Primary source 1

Cited in chapters: 1, 4, 19, 22, 23.

EX05. PersonaGym: Evaluating Persona Agents and LLMs

Vinay Samuel et al., Findings of EMNLP, November 2025

Findings of EMNLP, November 2025.

Primary source 1

Cited in chapters: 19.

EX08. Reflexion: Language Agents with Verbal Reinforcement Learning

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan and Shunyu Yao

NeurIPS 2023 proceedings.

Primary source 1

Cited in chapters: 12.

EX09. Agentic Context Engineering: Evolving Contexts for Self-Improving Language Models

Qizheng Zhang et al

arXiv v3, March 29, 2026; title page identifies ICLR 2026.

Primary source 1 Primary source 2

Cited in chapters: 12, 22, 23, 24.

EX10. GEPA: Reflective Prompt Evolution Can Outperform Reinforcement Learning

Lakshya A Agrawal et al.

arXiv:2507.19457v2, February 14, 2026; title page identifies ICLR 2026 Oral

Primary source 1 Primary source 2

Cited in chapters: 12.

EX11. Self-Adapting Language Models

Adam Zweiger et al

arXiv v2, September 18, 2025.

Primary source 1 Primary source 2

Cited in chapters: 12, 22, 23, 24.

EX13. A-MEM: Agentic Memory for LLM Agents

Wujiang Xu et al

arXiv v11, October 8, 2025; title page identifies NeurIPS 2025.

Primary source 1 Primary source 2

Cited in chapters: 2, 6.

FN02. Developing Time-Oriented Database Applications in SQL

Richard T. Snodgrass

Author lists July 1999, Morgan Kaufmann, ISBN 1-55860-436-7.

Primary source 1 Primary source 2

Cited in chapters: 9, 18.

HI02. AI can help people feel heard, but an AI label diminishes this impact

Yidan Yin; Nan Jia; Cheryl J. Wakslak

Published PNAS article, 121(14), e2319112121; issue dated April 2, 2024.

Primary source 1

Cited in chapters: 15.

HI03. AI Companions Reduce Loneliness

Julian De Freitas; Zeliha Oğuz-Uğuralp; Ahmet Kaan Uğuralp; Stefano Puntoni

Journal of Consumer Research 52(6), April 2026, pp. 1126-1148; corrected/typeset September 24, 2025.

Primary source 1

Cited in chapters: 16, 22, 23.

HI04. How AI and Human Behaviors Shape Psychosocial Effects of Extended Chatbot Use: A Longitudinal Randomized Controlled Study

Cathy Mengying Fang; Auren R. Liu; Valdemar Danry; Eunhae Lee; Samantha W. T. Chan; Pat Pataranutaporn; Pattie Maes; Jason Phang; Michael Lampe; Lama Ahmad; Sandhini Agarwal

arXiv v2; initial submission March 21, 2025; preprint.

Primary source 1

Cited in chapters: 2, 16, 22, 23.

HI05. How AI Companionship Develops: Evidence from a Longitudinal Study

Angel Hsing-Chi Hwang; Fiona Li; Jacy Reese Anthis; Hayoun Noh

arXiv v1; preprint.

Primary source 1

Cited in chapters: 16.

HI06. Toward Natural and Companionable Virtual Agents via Cross-Temporal Emotional Modeling

Feier Qin; Xiao Li; Yi Zheng; Haibin Huang; Hanyao Wang; Xiaoyu Wang; Yan Lu; Yuan Zhang

arXiv v1 author manuscript identifying the CHI 2026 paper; related DOI 10.1145/3772318.3790917.

Primary source 1

Cited in chapters: 11.

HI07. Training language models to be warm can reduce accuracy and increase sycophancy

Lujain Ibrahim; Franziska Sofia Hafner; Luc Rocher

Published Nature article, volume 652, pp. 1159-1165; accepted March 12, 2026.

Primary source 1

Cited in chapters: 15.

HI08, S05. Sycophantic AI decreases prosocial intentions and promotes dependence

Myra Cheng; Cinoo Lee; Pranav Khadpe; Sunny Yu; Dyllan Han; Dan Jurafsky

Final Science article, volume 391, issue 6792, aec8352. Supersedes the narrower October 2025 preprint dataset.

Primary source 1

Cited in chapters: 15.

HI09. On the conversational persuasiveness of GPT-4

Francesco Salvi; Manoel Horta Ribeiro; Riccardo Gallotti; Robert West

Published Nature Human Behaviour article, volume 9, pp. 1645-1653, read with September 3, 2026 author correction.

Primary source 1

Cited in chapters: 15.

HI10. Mourning the loss of AI companions

Julian De Freitas; Noah Castelo; Ahmet Kaan Uğuralp; Zeliha Oğuz-Uğuralp

Published Nature Human Behaviour article; supersedes earlier working-paper discussions of the Replika update.

Primary source 1

Cited in chapters: 16.

HI11. To Trust or to Think: Cognitive Forcing Functions Can Reduce Overreliance on AI in AI-assisted Decision-making

Zana Buçinca; Maja Barbara Malaya; Krzysztof Z. Gajos

Pinned arXiv v1; related 2021 PACM HCI publication DOI 10.1145/3449287.

Primary source 1

Cited in chapters: 15.

HI12. Humanlike AI Design Increases Anthropomorphism but Yields Divergent Outcomes on Engagement and Trust Globally

Robin Schimmelpfennig; Mark Díaz; Vinodkumar Prabhakaran; Aida Davani

arXiv v2; initial submission December 19, 2025; preprint.

Primary source 1

Cited in chapters: 15.

HI13. My Chatbot Companion - a Study of Human-Chatbot Relationships

Marita Skjuve; Asbjørn Følstad; Knut Inge Fostervold; Petter Bae Brandtzæg

International Journal of Human-Computer Studies 149, article 102601, May 2021.

Primary source 1

Cited in chapters: 16.

HI14. Emotional Manipulation by AI Companions

Julian De Freitas; Zeliha Oğuz-Uğuralp; Ahmet Kaan-Uğuralp

arXiv v3; initial submission August 15, 2025; preprint.

Primary source 1

Cited in chapters: 16.

R4A01. LaMP: When Large Language Models Meet Personalization

Alireza Salemi; Sheshera Mysore; Michael Bendersky; Hamed Zamani

arXiv:2304.11406v4, June 5, 2024

Primary source 1 Primary source 2

Cited in chapters: 1, 2, 6.

R4A02. ReAct: Synergizing Reasoning and Acting in Language Models

Shunyu Yao; Jeffrey Zhao; Dian Yu; Nan Du; Izhak Shafran; Karthik Narasimhan; Yuan Cao

arXiv:2210.03629v3, March 10, 2023

Primary source 1 Primary source 2

Cited in chapters: 8.

R4A03, R4B02, R4C10. Evaluating Very Long-Term Conversational Memory of LLM Agents

Adyasha Maharana; Dong-Ho Lee; Sergey Tulyakov; Mohit Bansal; Francesco Barbieri; Yuwei Fang

ACL 2024 final, pp. 13851-13870; DOI 10.18653/v1/2024.acl-long.747

Primary source 1 Primary source 2

Cited in chapters: 2, 10, 20, 22, 23.

R4A04, R4B01, R4C13. LLM Agents Grounded in Self-Reports Enable General-Purpose Simulation of Individuals

Joon Sung Park; Carolyn Q. Zou; Jonne Kamphorst; Niles Egan; Aaron Shaw; Benjamin Mako Hill; Carrie Cai; Meredith Ringel Morris; Percy Liang; Robb Willer; Michael S. Bernstein

arXiv:2411.10109v3, June 28, 2026

Primary source 1 Primary source 2

Cited in chapters: 2, 14, 24.

R4A05, R4C01. The Protection of Information in Computer Systems

Jerome H. Saltzer; Michael D. Schroeder

Proceedings of the IEEE 63(9), 1975, pp. 1278-1308; author-hosted HTML reproduction, Part I

Primary source 1 Primary source 2

Cited in chapters: 5, 17.

R4A06. Enabling Large Language Models to Generate Text with Citations

Tianyu Gao; Howard Yen; Jiatong Yu; Danqi Chen

EMNLP 2023, pp. 6465-6488; DOI 10.18653/v1/2023.emnlp-main.398

Primary source 1 Primary source 2

Cited in chapters: 5, 8.

R4A07. PostgreSQL 18 Documentation: 3.4 Transactions

PostgreSQL Global Development Group

PostgreSQL 18 documentation, accessed October 15, 2026

Primary source 1

Cited in chapters: 7.

R4B03. Lost in the Middle: How Language Models Use Long Contexts

Nelson F. Liu et al.

arXiv:2307.03172v3, November 20, 2023

Primary source 1 Primary source 2

Cited in chapters: 10.

R4B04. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks

Patrick Lewis et al.

arXiv:2005.11401v4, April 12, 2021

Primary source 1 Primary source 2

Cited in chapters: 10.

R4B05. Sleep-time Compute: Beyond Inference Scaling at Test-time

Kevin Lin; Charlie Snell; Yu Wang; Charles Packer; Sarah Wooders; Ion Stoica; Joseph E. Gonzalez

arXiv:2504.13171v1, April 17, 2025

Primary source 1 Primary source 2

Cited in chapters: 13.

R4B06. Time, Clocks, and the Ordering of Events in a Distributed System

Leslie Lamport

Communications of the ACM 21(7), July 1978, pp. 558-565

Primary source 1

Cited in chapters: 13.

R4B07. The Chubby Lock Service for Loosely-Coupled Distributed Systems

Mike Burrows

OSDI 2006 paper

Primary source 1 Primary source 2

Cited in chapters: 13.

R4B08. Learning from Time-Changing Data with Adaptive Windowing

Albert Bifet; Ricard Gavaldà

Author manuscript dated October 17, 2006; related publication SDM 2007

Primary source 1 Primary source 2

Cited in chapters: 12.

R4B09. ALMA: A Layered Model of Affect, project documentation

Patrick Gebhard / DFKI

Author project documentation accessed October 15, 2026; cites AAMAS 2005 model paper

Primary source 1

Cited in chapters: 11.

R4B10. Machine Unlearning

Lucas Bourtole et al.

arXiv:1912.03817v3, December 15, 2020; metadata identifies IEEE S&P 2021 publication

Primary source 1

Cited in chapters: 9.

R4B11. IMPersona: Evaluating Individual Level LM Impersonation

Quan Shi; Carlos E. Jimenez; Stephen Dong; Brian Seo; Caden Yao; Adam Kelch; Karthik Narasimhan

arXiv:2504.04332v2, April 8, 2025

Primary source 1 Primary source 2 Primary source 3

Cited in chapters: 14.

R4B12. Whose Opinions Do Language Models Reflect?

Shibani Santurkar; Esin Durmus; Faisal Ladhak; Cino Lee; Percy Liang; Tatsunori Hashimoto

arXiv:2303.17548v1, March 30, 2023

Primary source 1

Cited in chapters: 14.

R4C02. The Off-Switch Game

Dylan Hadfield-Menell; Anca Dragan; Pieter Abbeel; Stuart Russell

IJCAI 2017; arXiv:1611.08219

Primary source 1

Cited in chapters: 17.

R4C03. AgentDojo: A Dynamic Environment to Evaluate Prompt Injection Attacks and Defenses for LLM Agents

Edoardo Debenedetti et al.

arXiv:2406.13352v3

Primary source 1

Cited in chapters: 17, 23.

R4C04. Equivalence Tests: A Practical Primer for t Tests, Correlations, and Meta-Analyses

Daniël Lakens

2017; Social Psychological and Personality Science 8(4), 355-362; DOI 10.1177/1948550617697177

Primary source 1

Cited in chapters: 21, 23.

R4C05. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena

Lianmin Zheng et al.

arXiv:2306.05685v4

Primary source 1

Cited in chapters: 19.

R4C06. Regulation (EU) 2016/679 (General Data Protection Regulation)

European Parliament and Council of the European Union

Official text, 27 April 2016, selected provisions checked 2026-10-15

Primary source 1

Cited in chapters: 18, 21, 24.

R4C07. Regulation (EU) 2024/1689 (Artificial Intelligence Act)

European Parliament and Council of the European Union

Consolidated 27 July 2026, CELEX:02024R1689-20260727

Primary source 1 Primary source 2

Cited in chapters: 18, 24.

R4C08. Authorization - Model Context Protocol

Model Context Protocol contributors

Specification version 2025-11-25

Primary source 1

Cited in chapters: 21, 24.

R4C09. A2A Protocol Specification

A2A Protocol Working Group

Live latest specification accessed 2026-10-15; page is mutable, no immutable release inferred

Primary source 1

Cited in chapters: 21, 24.

R4C11. Lost in the Middle: How Language Models Use Long Contexts

Nelson F. Liu; Kevin Lin; John Hewitt; Ashwin Paranjape; Michele Bevilacqua; Fabio Petroni; Percy Liang

TACL 12 (2024), 157-173; DOI 10.1162/tacl_a_00638

Primary source 1 Primary source 2

Cited in chapters: 24.

R4C12. Memory in ChatGPT

OpenAI

Help Center article accessed October 15, 2026.

Primary source 1

Cited in chapters: 24.

RG01. First-in-the-Nation AI Chatbot Safeguards Signed into Law

California State Senator Steve Padilla, official Senate website

October 13, 2025 enactment announcement for SB 243; stated general effective date January 1, 2026.

Primary source 1 Primary source 2

Cited in chapters: 18.

RG02. 人工智能拟人化互动服务管理暂行办法

Cyberspace Administration of China and four other issuing authorities

Order No. 21, published April 10, 2026; effective July 15, 2026.

Primary source 1

Cited in chapters: 18, 24.

S02. EMA: A process model of appraisal dynamics

Stacy C. Marsella; Jonathan Gratch

Cognitive Systems Research 10(1), 70-90; available online 2008-07-05; DOI 10.1016/j.cogsys.2008.03.005

Primary source 1

Cited in chapters: 11.

S03. From Triggers to Emotions: A CPM-Grounded Appraisal Multi-Agent for Dynamic Emotional Evolution in Persona-Based Dialogue

Jingyao Cai; Shuaijun Liu; Abdul Rehman; Yutong Guo; Qin Tian; Thomas Dolby; Sue Green; Chantel Cox; Xiaosong Yang

arXiv:2607.07824v1

Primary source 1

Cited in chapters: 11.

S04. PAIR: Perceptual Affective Inference and Regulation in a Real-Time Multimodal Conversational Agent

Kexin Quan; Zijian Ding; Jiaye Yong; Qinshi Zhang; Dong Wang; Jessie Chin

arXiv:2610.07523v1

Primary source 1

Cited in chapters: 2.

S06. Sycophantic AI makes human interaction feel more effortful and less satisfying over time

Lujain Ibrahim; Franziska Sofia Hafner; Myra Cheng; Cinoo Lee; Rebecca Anselmetti; Robb Willer; Luc Rocher; Diyi Yang

arXiv:2605.07912v3; first submission 2026-05-08

Primary source 1

Cited in chapters: 16.

S08. Defeating Prompt Injections by Design

Edoardo Debenedetti; Ilia Shumailov; Tianqi Fan; Jamie Hayes; Nicholas Carlini; Daniel Fabian; Christoph Kern; Chongyang Shi; Andreas Terzis; Florian Tramèr

arXiv:2503.18813v2; published at SaTML 2026, DOI 10.1109/SaTML68715.2026.00040

Primary source 1

Cited in chapters: 8, 17.

S09. Can CaMeLs Talk? Securing Multi-Agent Systems Against Indirect Prompt Injection Attacks

James Peters-Gill; Avi Semler; Henning Bartsch; Ilia Shumailov; Christian Schroeder de Witt
arXiv:2610.05640v1

Primary source 1

Cited in chapters: 17.

S10. Incomplete Tasks Induce Shutdown Resistance in Some Frontier LLMs

Jeremy Schlatter; Benjamin Weinstein-Raun; Jeffrey Ladish

arXiv:2509.14260v2; first submission 2025-09-13; TMLR January 2026 status reported by authors

Primary source 1

Cited in chapters: 17.

S11. Safely Interruptible Agents

Laurent Orseau; Stuart Armstrong

UAI 2016 proceedings paper

Primary source 1

Cited in chapters: 17.

TX01. Cognitive Architectures for Language Agents (CoALA)

Sumers et al.

2024-03-15; arXiv v3

Primary source 1

Cited in chapters: 1, 2, 6, 21.

TX02. Voyager: An Open-Ended Embodied Agent with Large Language Models

Wang et al.

2023; arXiv 2305.16291

Primary source 1 Primary source 2

Cited in chapters: 2, 6, 12, 24.

TX09. PostgreSQL 18: Row Security Policies

PostgreSQL Global Development Group

Version 18; retrieved 2026-10-15

Primary source 1

Cited in chapters: 7, 8, 17.

TX10. PostgreSQL 18: Transaction Isolation

PostgreSQL Global Development Group

Version 18; retrieved 2026-10-15

Primary source 1

Cited in chapters: 7, 13.

TX11. pgvector

pgvector contributors

Retrieved 2026-10-15; rolling repository README

Primary source 1

Cited in chapters: 7.

TX17. A cognitive-affective system theory of personality

Mischel and Shoda

1995; Psychological Review 102(2)

Primary source 1

Cited in chapters: 4.

TX18. Whole Trait Theory

Fleeson and Jayawickreme

2015; Journal of Research in Personality 56

Primary source 1 Primary source 2

Cited in chapters: 4.

TX23. New York General Business Law, Article 47: AI Companion Models

New York State

Official Sections 1700-1702, most recent revision displayed November 7, 2025; accessed October 15, 2026

Primary source 1 Primary source 2 Primary source 3

Cited in chapters: appendices or figures.

TX24. Guidelines on AI transparency obligations and Article 50 FAQ

European Commission

Live guidance checked 2026-10-15

Primary source 1 Primary source 2

Cited in chapters: 18.

TX25. Memory Injection Attacks on LLM Agents via Query-Only Interaction

Dong et al.

2026-02-12; arXiv v5

Primary source 1

Cited in chapters: 20, 22.

Companion applications and materials

The companion page brings together the free digital edition, three local applications and their source code, the teaching laboratory, full-size figures and the source-reading index.

Persona keeps an authored character and shared memory. **Home Repair Advisor** organizes a repair case and its evidence. **Mara** records experience and permits bounded changes to habits and views.

The applications run locally with Node.js 24 or later and use the model API configured by the reader. Downloading the software is free; model providers may charge for usage. Setup instructions and the limits of each implementation accompany the code.



<https://denisostapenko.com/books/ai-personas/companion>

<https://github.com/denis-ostapenko/ai-personas>

Denis Ostapenko



denisostapenko.com