Whiskers v1.1

Local-First AI Continuity and Context Management

Whitepaper

Abstract

Large Language Models have demonstrated remarkable reasoning capabilities, yet they remain constrained by a fundamental limitation: continuity.

As conversations grow longer, important information becomes increasingly difficult to maintain. Sessions end, context is lost, and users are often required to reconstruct prior discussions, project knowledge, assumptions, and decisions.

Whiskers addresses this problem through explicit, user-controlled continuity.

Rather than attempting to create a model with permanent memory, Whiskers treats the language model as a stateless reasoning engine and provides continuity through persistent context artifacts that remain visible, editable, and under user control.

The core thesis of Whiskers is simple:

The model reasons. Whiskers remembers.

Whiskers is a local-first AI continuity and context-management system designed to support long-running AI-assisted workflows through persistent context, user-maintained knowledge, and correction-driven memory management.

---

1. Problem Statement

Modern AI systems answer one question exceptionally well:

> What should be said next?

They answer a different question much less effectively:

> What have we been doing for the last month?

As conversations grow, users encounter familiar challenges:

- Lost context
- Forgotten decisions
- Repeated explanations
- Session resets
- Inconsistent continuity
- Dependence on provider-managed memory systems

Projects, stories, investigations, research efforts, and long-term planning frequently span weeks or months. Yet most AI systems operate primarily within the boundaries of a single conversation.

The result is a recurring pattern:

The user remembers.

The model forgets.

Whiskers was created to address this gap.

---

2. What Whiskers Stores

A common misconception is that Whiskers is an AI model.

It is not.

Whiskers does not perform reasoning.

The language model performs reasoning.

Whiskers provides continuity.

To accomplish this, Whiskers maintains four persistent context sources.

Contract

The Contract defines operating rules and behavioral constraints.

Examples include:

- Writing requirements
- Project rules

- Character behavior constraints
- Organizational standards
- User preferences

The Contract answers:

> How should the model behave?

Canon

The Canon contains information expected to remain true across sessions.

Examples include:

- Project architecture
- Character information
- World-building data
- Product requirements
- Established facts

The Canon answers:

> What should the model know?

Session Log

The Session Log records interaction history.

Recent portions of the log are reintroduced into future prompts to provide short-term continuity.

The Session Log answers:

> What has recently happened?

Summaries

Summaries provide compressed representations of long-running sessions.

They allow important information to survive beyond the limits of active conversational context.

Summaries answer:

> What happened before the current session?

Together these artifacts form the continuity layer surrounding an otherwise stateless reasoning engine.

Component	Purpose
Contract	Behavioral constraints
Canon	Persistent facts
Session Log	Recent continuity
Summaries	Long-term continuity

Whiskers therefore stores context rather than intelligence.

---

3. Design Philosophy

Whiskers is built around several principles.

3.1 Model Agnostic

Whiskers is independent of any specific language model.

Reasoning engines can be replaced without altering the continuity architecture.

Models evolve.

Continuity remains.

3.2 Local First

Whiskers is designed to operate locally.

User knowledge remains under user control.

No cloud infrastructure is required.

No external memory service is required.

All continuity artifacts are stored as ordinary files.

3.3 Explicit Context

Every continuity source is visible.

Every continuity source is editable.

Every continuity source can be inspected using ordinary tools.

Whiskers favors transparency over hidden memory systems.

3.4 User Authority

The user is the final source of truth.

Not the model.

Not the runtime.

Not the summarizer.

Whiskers assumes that users possess domain knowledge the model may not.

The system therefore provides mechanisms for correction before information becomes part of future continuity.

---

4. Architecture

Whiskers operates as a continuity layer between the user and the language model.

```
User
    ↓
Prompt Assembly
    ↓
Contract
Canon
Session Log
Summaries
    ↓
Language Model
    ↓
```

```
Response
    ↓
Accept / Rewrite
    ↓
Persistent Storage
```

Each component has a clearly defined responsibility.

Component	Responsibility
Language Model	Reasoning
Runtime	Context assembly
Contract	Constraints
Canon	Ground truth
Session Log	Short-term continuity
Summaries	Long-term continuity

This separation allows reasoning and continuity to evolve independently.

---

5. Context Model

Whiskers treats continuity as a reconstruction problem.

Each prompt is assembled from multiple context sources.

The resulting context may contain:

- User input
- Contract
- Canon
- Session history
- Session summaries

The model receives all relevant context during prompt construction.

No assumption is made that the model remembers prior interactions.

Whiskers intentionally treats the language model as stateless.

Continuity is achieved through repeated context reconstruction rather than assumed model memory.

As conversations grow, older interactions naturally move beyond the active context window.

To address this limitation, Whiskers combines:

- Sliding-window history
- Persistent canon data
- Long-term summaries
- User-directed knowledge management

Important information therefore remains available even when individual sessions exceed practical context limits.

---

6. Rewrite and Correction

A defining feature of Whiskers is the Rewrite workflow.

Most AI systems operate as follows:

```
Model Output
      ↓
Stored History
```

Whiskers introduces an approval step.

```
Model Output
      ↓
User Review
      ↓
Accept or Rewrite
      ↓
Stored History
```

The model’s initial response is not automatically treated as authoritative.

Users may correct:

- Hallucinations
- Continuity errors
- Incorrect assumptions
- Project mistakes
- Narrative inconsistencies

When a rewrite occurs, the corrected version becomes the version preserved within continuity.

This allows users to actively manage information quality over long-running sessions.

Continuity is therefore based upon user-approved information rather than model-generated information alone.

---

7. Validation Status

Whiskers has been validated through practical long-duration usage and multi-model testing.

Demonstrated capabilities include:

- Long-running roleplaying campaigns
- Persistent world-building projects
- Cross-session continuity
- Character persistence
- Project knowledge retention
- Canon-driven correction
- Rewrite-driven correction

The architecture has been tested across multiple hardware configurations ranging from modest CPU-only systems to larger GPU-assisted systems.

Multiple language models have successfully operated within the same continuity framework.

These demonstrations validate the continuity architecture rather than any individual model.

---

8. Known Limits

Whiskers intentionally does not attempt to solve several problems.

These include:

- Model intelligence
- Truth verification
- Autonomous knowledge management
- Automatic fact checking
- Team collaboration
- Distributed synchronization
- Enterprise knowledge governance
- Automatic canon generation

Whiskers improves continuity.

It does not guarantee correctness.

A model may still be wrong.

Users remain responsible for maintaining authoritative knowledge sources.

---

9. Future Directions

Potential future development areas include:

- Mobile access
- Simplified installation
- Enhanced summary generation
- Multi-session management
- Shared continuity environments
- Additional context inspection tools
- Expanded model support
- Enterprise deployment options

Future capabilities will build upon the existing continuity architecture rather than replace it.

---

10. Conclusion

Whiskers does not attempt to make AI smarter.

It attempts to make AI continuous.

The project's contribution is not a new language model, inference engine, or reasoning architecture.

Its contribution is a user-controlled continuity layer that surrounds existing models and preserves knowledge across sessions.

By combining Contracts, Canon files, Session Logs, Summaries, and user-directed correction, Whiskers provides a practical framework for maintaining continuity in long-running AI-assisted workflows.

Its central proposition remains:

The model reasons.

Whiskers remembers.

Why This Matters

Traditional AI:

```
Start over.
```

Whiskers:

```
Continue.
```

Traditional AI:

```
Trust the model to remember.
```

Whiskers:

```
Reconstruct context explicitly.
```

Traditional AI:

```
Memory is opaque.
```

Whiskers:

```
Continuity is visible and editable.
```

Traditional AI:

```
The model owns continuity.
```

Whiskers:

```
The user owns continuity.
```

Appendix A: AI Concepts

Language Model

A language model generates text by predicting likely continuations from provided input.

Whiskers does not replace the language model.

Whiskers provides continuity around it.

Context Window

A language model can only consider a finite amount of information at one time.

This limit is known as the context window.

As new information enters the context window, older information eventually falls outside it.

Whiskers addresses this limitation through persistent context sources and summarization.

Tokens

Language models process text as tokens rather than words.

A token is typically a word fragment or short text segment.

Context windows are commonly measured in tokens.

Hallucination

A hallucination occurs when a model confidently generates information that is incorrect, unsupported, or fabricated.

Hallucinations are a normal characteristic of probabilistic language models because they generate text by statistically selecting likely next tokens rather than by understanding, verifying, or reasoning about the information they produce. As a result, a model may generate outputs that appear coherent and confident even when they are inaccurate or entirely invented.

Whiskers does not eliminate hallucinations.

It provides tools that allow users to identify and correct them before they become part of long-term continuity.

Prompt

A prompt is the information provided to a language model before it generates a response.

In Whiskers, prompts may include user input, contracts, canon information, session history, and summaries.

Stateless Model

For purposes of Whiskers, a model is treated as stateless.

This means the system assumes the model retains no guaranteed memory between interactions.

Continuity is reconstructed from stored context rather than assumed model memory.

Summary Model

A summary model is a secondary language model used to condense long-running sessions into shorter representations.

These summaries preserve important information while reducing context size.

Appendix B: Frequently Asked Questions

Doesn’t ChatGPT already have memory?

Some hosted AI platforms provide memory features.

Whiskers differs in that continuity is explicit, visible, editable, portable, and independent of any particular provider.

Why not simply use larger context windows?

Larger context windows increase capacity but do not eliminate context limits.

All practical systems eventually encounter finite context constraints.

Whiskers focuses on continuity management rather than context expansion.

Is Whiskers a vector database?

No.

Whiskers uses explicit user-managed context artifacts rather than semantic retrieval systems.

Is Whiskers an AI model?

No.

Whiskers is a continuity and context-management system that operates alongside a language model.

Why local-first?

Local operation provides transparency, portability, privacy, and independence from external services.

Does Whiskers guarantee correctness?

No.

Whiskers improves continuity.

Users remain responsible for validating facts and maintaining authoritative knowledge.

Can continuity files be edited manually?

Yes.

Contracts, Canon files, Session Logs, and Summaries are ordinary text files intended to remain visible and editable.

This behavior is intentional and aligns with Whiskers' emphasis on transparency and user control.