IRIS

ENGINEERING A LOCAL-FIRST PERSONAL INTELLIGENCE SYSTEM

SEPTEMBER, 2026

RESEARCH QUESTION

—

HOW CAN A PERSONAL AI ASSISTANT EVOLVE FROM A PROMISING PROTOTYPE INTO A TRUSTWORTHY EVERYDAY SYSTEM WITHOUT SACRIFICING PRIVACY, EVIDENCE, OR USER CONTROL?

MOSES YOON

pyoon@uchicago.edu

ABSTRACT

Iris is a local-first personal intelligence system built to test a deceptively difficult proposition: *whether an AI assistant can be useful across everyday work while remaining honest about its evidence, permissions, and limits*.

This paper follows the complete development arc from v0.0—the product thesis of a persistent conversational presence—through the local Core, voice, current-information, document, Desktop, integration, action, Quality, and recovery work that culminated in AXIOM/Iris v0.11.12.

It reconstructs the product vision, architecture, interface evolution, release sequence, recurring failures, remediation strategy, and current readiness using versioned project records, qualification evidence, public-safe screenshots, and primary research on human–AI interaction, retrieval-augmented generation, agentic tool use, local-first software, speech recognition, and security.

The result is not a claim of general intelligence or a peer-reviewed efficacy study. It is an engineering case study about turning a compelling interface into an accountable system. The central finding is that assistant quality depends less on the number of integrations than on five linked properties: authority, freshness, provenance, continuity, and recoverability. v0.11.12 closes the engineering scope of the 0.11 family, but owner-assisted acoustic evidence, provider breadth, safe live-action qualification, and longitudinal daily-use evidence remain incomplete.

The forward programme therefore extends through v0.12 research, v0.13 financial analysis, v0.14 composed workflows, v0.15 compute and voice architecture, v0.16 hardening, a frozen v1.0 candidate, and separately governed post-v1 device, perception, and ambient-computing work.

Keywords: local-first AI; personal assistant; grounded retrieval; human–AI interaction; voice interface; tool use; provenance; fail-closed systems; evaluation

DISCLOSURE AND AUTHORSHIP

I, Moses Yoon, lead the product vision, requirements, interaction design, testing priorities, and release strategy. Engineering implementation and verification are AI-assisted. This document intentionally excludes credentials, private communications, proprietary prompts, and implementation details that could enable replication.

TABLE OF CONTENTS

1. INTRODUCTION

Why a Personal Assistant is a Systems Problem

The modern assistant demo is usually framed as a language problem: ask a question, receive a fluent answer, and judge whether it sounds intelligent. Daily use reveals a harder reality. A useful assistant must decide which source is authoritative, determine whether that source is current, preserve what the user is referring to across turns, request permission before acting, recover from interruption, and communicate uncertainty without becoming robotic. The answer is only the visible end of a much larger system.

Iris began with a familiar ambition—a desktop presence that could converse, speak, search personal material, check calendars and email, and help organise work. The goal was not to imitate a science-fiction interface literally. It was to capture the quality that makes fictional assistants compelling: continuity. The assistant should feel as though it understands the current situation, can reach the right tools, and remains available without making the user manage the machinery.

That ambition immediately creates a tension. The more domains Iris can reach, the more damaging a confident mistake becomes. A wrong general-knowledge answer is irritating. A fabricated appointment, stale inbox claim, incorrect document, or unapproved external action can break trust. NIST's AI Risk Management Framework treats trustworthiness as a lifecycle concern rather than a property of a model alone [1]; human–AI interaction research similarly stresses making capabilities, limitations, corrections, and recovery legible to users [3]. Iris therefore evolved around a core proposition: capability must be earned through evidence and control.

RESEARCH QUESTION

How can a personal AI assistant evolve from a promising prototype into a trustworthy everyday system without sacrificing privacy, evidence, or user control?

The paper answers this question as a longitudinal engineering case study. It connects product decisions to observed failures, shows how the architecture changed, reports both automated and live-test evidence, and states what remains unfinished. This candid framing matters for a public portfolio: the project is most interesting not because every feature is complete, but because repeated failures were converted into explicit design constraints.

2. THE ORIGINAL VISION

2.1 A local presence, not another dashboard

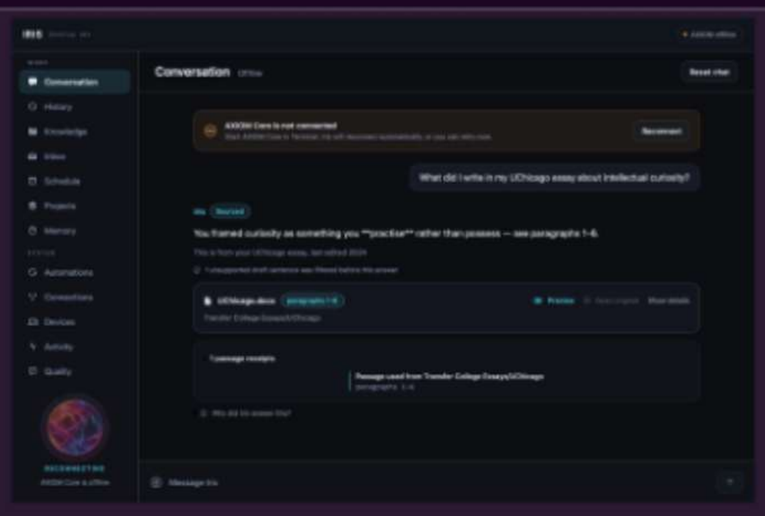
The first concept centred on a persistent Iris presence: a luminous visual object that communicates standby, listening, thinking, speaking, and interruption states. It serves as more than branding. Presence gives the assistant a stable spatial identity while text and controls change around it. The design direction remained minimal, futuristic, and professional—dark surfaces, restrained cyan and violet signals, clear hierarchy, and no decorative motion that obscures state.

The interaction model was equally important. Typing and voice were intended to be two entrances to the same conversation, not separate products. Clicking Iris could enter or leave voice interaction; the assistant should not jump left or right when modes change; and text should remain available during voice. Those details embody a broader rule: the interface should reduce mode management. The user should think about the request, not about which subsystem is active.

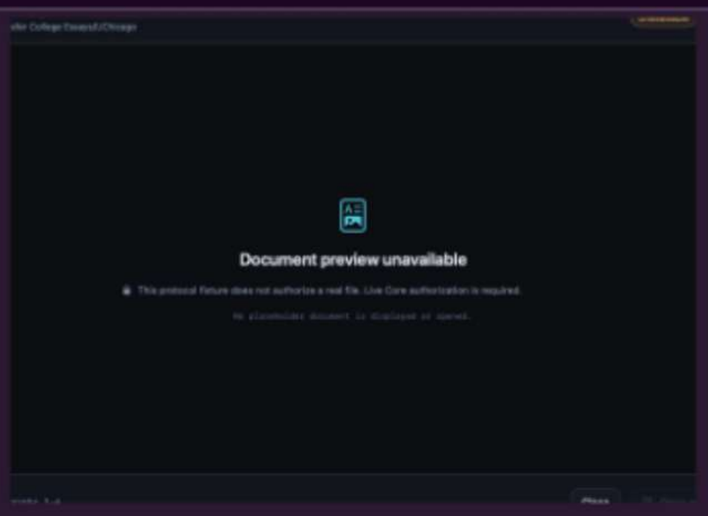
INTERFACE EVIDENCE

Iris as an accountable workspace

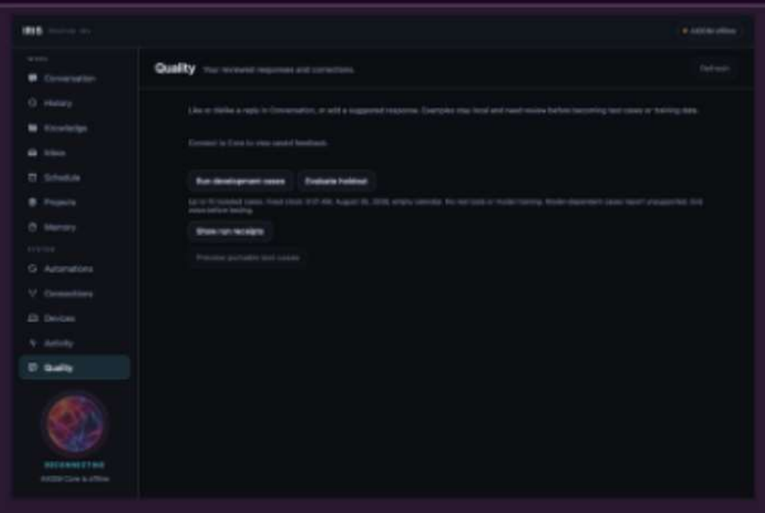
Current public-safe Desktop captures; content shown is synthetic development data.



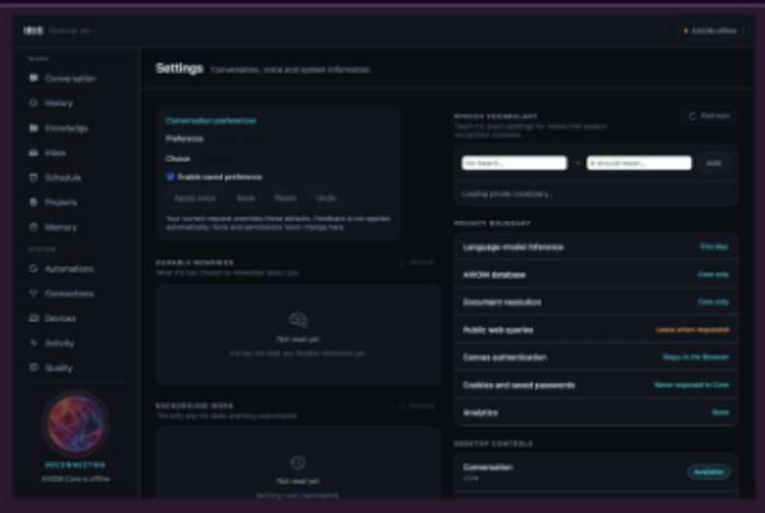
01 Grounded conversation



02 Document passage and receipt



03 Reviewable Quality Corpus



04 Visible privacy boundaries

Figure 1. The Iris workspace preserves a consistent visual language across conversation, evidence, quality review, and settings. Screenshots contain synthetic development data.

2.2 From Personal organiser to general reasoning partner

The long-term vision extends beyond schedules and inboxes. Iris should locate and compare documents, understand coursework, search the web, analyse companies, help draft work, maintain preferences, and eventually coordinate bounded actions. It should also recognise when a request is personal, factual, creative, operational, or emotionally conversational. A request such as “I've had a rough day—no advice, just some company” requires a different response policy from “show me that transfer essay” or “what do I have next week?”

The project deliberately rejected two shortcuts. First, a broad feature list would not substitute for reliability. Second, “local-first” would not mean pretending every operation must happen offline. Local-first software is defined by ownership, availability, and user control, while still allowing collaboration and external services where appropriate [6]. Iris uses that principle as an architectural boundary: personal indexes, memory, feedback, and orchestration remain owner-controlled; remote providers or models are optional authorities whose use must be visible.

3. METHOD AND EVIDENCE

3.1 Development record

The project record includes versioned source code, release-family reports, architecture and constitution documents, decision logs, regression suites, public-safe preview images, and a private live qualification conducted against v0.11.9.5. The present case study verifies the installed Core and Desktop at v0.11.12 (Desktop build 62), protocol 1.23, and Browser Bridge 0.9.5 as of 4 September 2026. Browser Bridge is versioned separately because its lifecycle and permissions differ from the Desktop and Core.

Evidence layer	What it can support	What it cannot support
Source and unit tests	Implemented logic, regression protection, protocol contracts	Owner experience, provider freshness, audible latency
Packaged verification	Installed files and versions match the release	Correct behaviour in every live context
Protocol diagnostics	Tool path, payload, timing, controlled failures	Usable Desktop flow by itself
Desktop scenarios	Visible everyday behaviour and continuity	Acoustic claims without a microphone test
Longitudinal use	Repeatability across days, restarts, and changing data	Still incomplete for v0.11.12

Table 1. Evidence layers used in this case study.

3.2 Qualification standard

The live qualification was designed to answer a stricter question than “does the code pass?” Desktop interaction was the primary evidence path for connected calendar, email, documents, weather, web, and conversation. An isolated profile tested synthetic memories, editable objects, permission flows, cancellations, and controlled failures. Replayed recordings covered repeatable speech-recognition checks; however, actual owner-assisted microphone timing and interruption remained blocked.

Capabilities were assigned PASS, PARTIAL, FAIL, BLOCKED, NOT SAFELY TESTABLE, or NOT CURRENTLY ACTIVE. PASS required repeated live evidence, not code existence or one demonstration. Real connected-data checks were read-only. Mutations were qualified only in isolated resources or at the proposal/rejection stage. This distinction follows a general principle in evaluation: evidence must match the claim. A unit test can prove a parser branch; it cannot prove that a voice turn feels interruptible in a noisy room.

3.3 External framework

External literature was used to interpret the engineering record, not to retroactively claim scientific validation. Retrieval-augmented generation motivates separating model reasoning from retrieved evidence [4]. ReAct and Toolformer demonstrate how language models can interleave reasoning with tools [5], [20]. Amershi et al. provide interaction guidance for communicating capabilities and supporting correction [3]. NIST and OWASP frame risk, prompt injection, and trust as end-to-end system concerns [1], [2], [18]. Apple, Microsoft, Chrome, Canvas, and W3C documentation anchor implementation constraints around speech, OAuth, extension lifecycles, pagination, and accessibility [8]–[17].

4. ARCHITECTURE

Intelligence Inside Boundaries

Iris is best understood as a stack of mediated intelligence. Desktop is the visible workspace. AXIOM Core handles intent, context, source selection, planning, verification, and response policy. Provider and local tools expose specific authorities. Trust controls sit across the stack rather than at one final filter. This differs from a chatbot with plugins: the system records why it chose a source, what coverage it offered, and whether the operation is read-only, proposed, or executable.

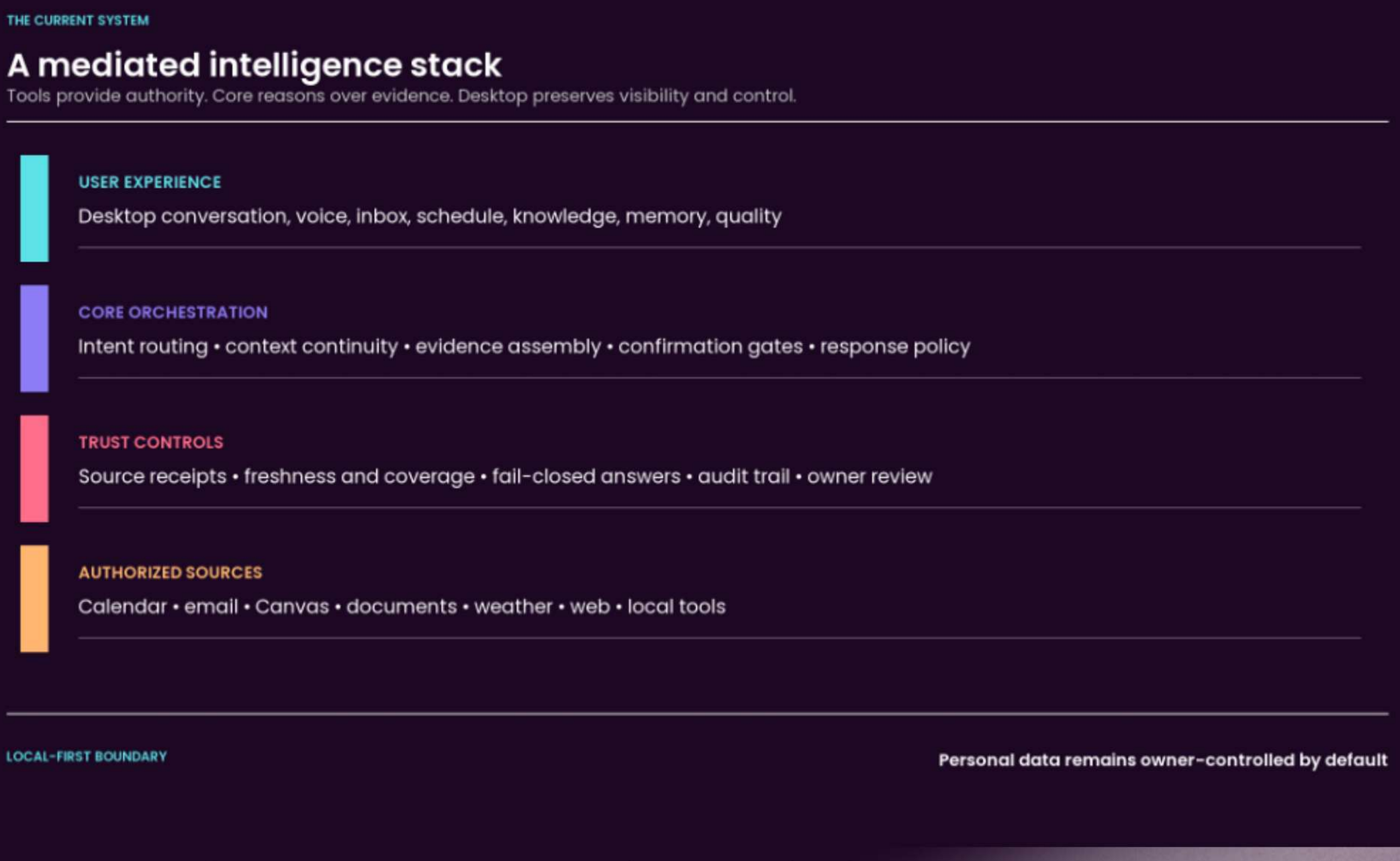


Figure 2. Current conceptual architecture. The intelligence layer is bounded by explicit source authority and owner-facing controls.

4.1 Authority, coverage, freshness, and health

By v0.11.10, connected sources were described through a source-state contract: authority identifies what the source can legitimately answer; coverage describes the subset available; freshness records when that subset was synchronised; and health indicates whether retrieval is operating normally. This language emerged from repeated email and Canvas problems. The Browser Bridge could expose only a bounded viewport or recent sample, yet early responses sounded as though the full account had been searched. The correction was not a more eloquent disclaimer. It was a structural model of the source's limits.

4.2 Evidence before prose

The answer pipeline therefore begins with interpretation and authorisation, not generation. The model may summarise, compare, or reason only after the relevant records and their state are available. If the evidence is missing, stale, or ambiguous, the assistant should ask, narrow the scope, or state the gap. This reflects RAG's separation between parametric memory and retrieved evidence [4], but extends it with operational controls: the evidence must also be authorised, current enough for the question, and presented with provenance.

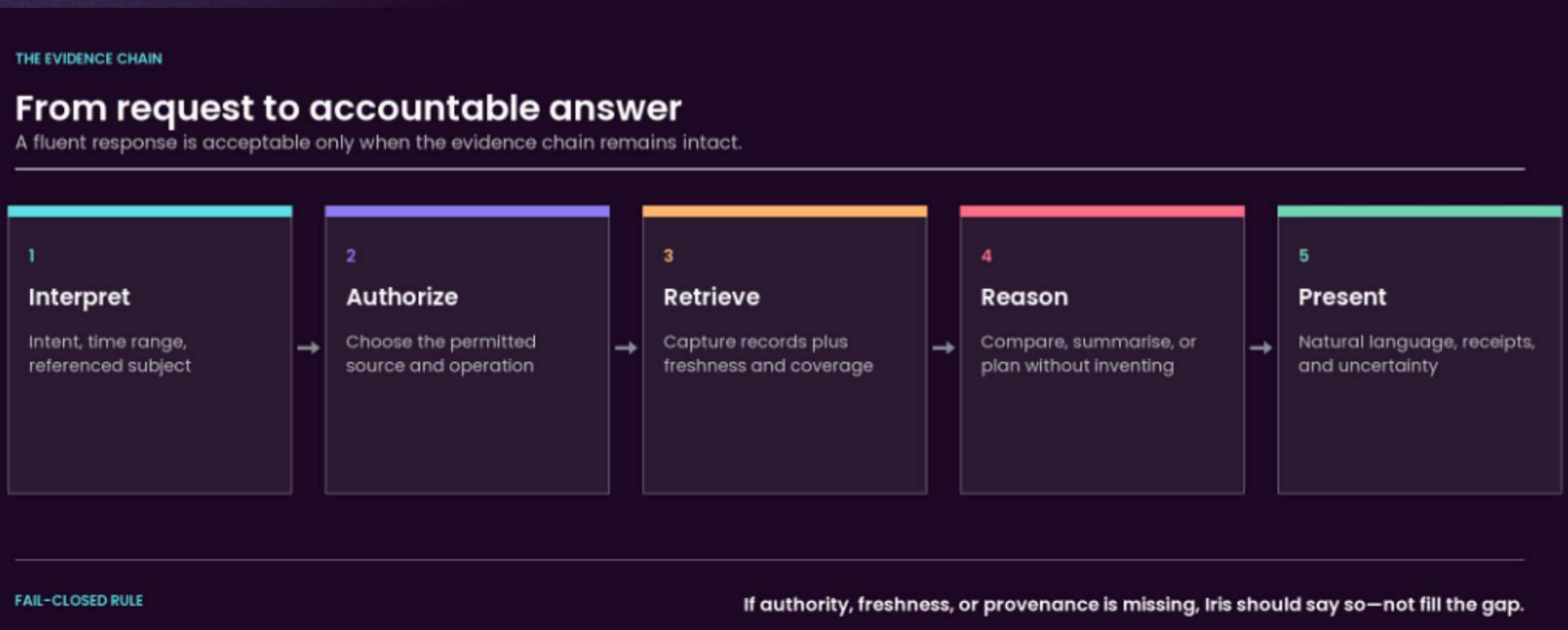


Figure 3. The evidence-to-answer pipeline. Natural language is the final layer, not the source of truth.

4.3 Confirmation-gated action

Read and write authority are intentionally separate. A request can be transformed into a proposed email, calendar change, reminder, or task while execution remains blocked behind review. At the v0.9.23 checkpoint, fourteen mutating tools required confirmation. This did not yet qualify every live provider action; it established the safety architecture. The design aligns with zero-trust reasoning: access is evaluated per resource and operation rather than assumed because the assistant is running locally [19].

5. DEVELOPMENT FROM V0.0 TO V0.11.12

The version history is a record of changing definitions of usefulness. v0.0 described a persistent, conversational Iris. v0.1–v0.4 established a local reasoning core and first voice and information paths. v0.5–v0.8 introduced measured reliability, grounded documents, an operations ledger, a native Desktop, and connected sources. v0.9 broadened daily work and safe actions. v0.10 made ordinary language and continuity release requirements. v0.11 turned qualification, source state, latency, provenance, and recovery into product surfaces. The grouped chronology in the appendix accounts for every release era while keeping the main argument readable.

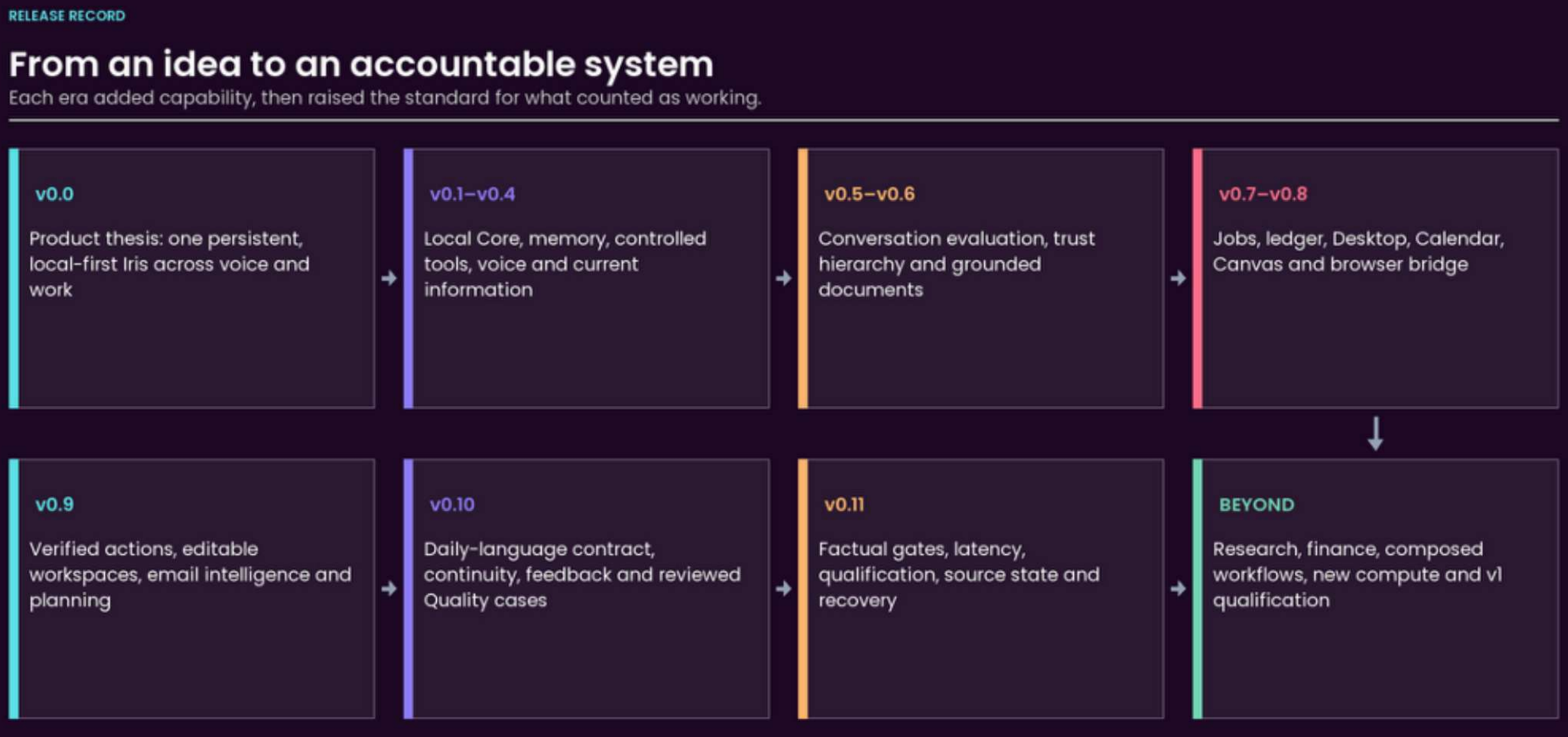


Figure 4. The full development arc from the v0.0 product thesis through v0.11.12 and the planned research-to-v1 programme.

Figure 5. Capabilities accumulated across release families. Later work extends rather than discards earlier authority and evidence boundaries.

5.1 v0.0: the pre-code product thesis

v0.0 was not a tagged software release. It was the design proposition that preceded implementation: Iris would be a stable presence rather than another collection of app windows; voice and text would share one conversation; personal context would remain private by default; and useful action would not require the owner to remember a command language. The luminous Iris presence, minimal dark interface, Work and System mental model, and ambition to span calendar, email, documents, coursework, research, and later devices all originate here. This concept also established the project's central tension: broad reach is desirable only if evidence, consent, and recovery grow with it.

5.2 v0.1: local intelligence, memory, and controlled tools

The first tagged family converted the concept into a provider-neutral local Core. v0.1.0 added the reproducible Python environment, Ollama-backed Brain interface, terminal conversation loop, SQLite conversation history, selective durable memory, controlled observation tools, and permission-gated mutation. v0.1.1 streamed responses, added a fast mode, bounded active context, and recorded local generation timing. The architectural choice was already clear: the model could propose and phrase, but the application would own tools, permissions, storage, and confirmation.

5.3 v0.2–v0.3: voice becomes an interaction system

v0.2 introduced time-bounded local recording, Whisper transcription, speech output, owner-only temporary audio, and timing records that excluded transcript content. The following patches explored a local neural identity, British and custom profiles, pitch, speed, expressiveness, and memory hand-off between language and voice workers on an 8 GB machine. v0.3 added repeated listen-think-speak conversation, silence endpointing, spoken exits, interruption, ear-oriented response planning, and bounded spoken context. These releases revealed that voice quality is not a single model choice; it is the combined behaviour of capture, endpointing, recognition, wording, synthesis, playback, and cancellation.

5.4 v0.4: current information and imperfect input

v0.4 added explicit-intent read-only web search, bounded public page reading, native Mac location, conversational lookups, and context-aware typed interaction. Subsequent patches hardened turn understanding, talk exits, practical-request inference, latency, private speech recognition, calibration, and personalised correction of imperfect input. The family ended with a recorded conversation baseline rather than a claim that natural language had been solved. Its lasting lesson was that a useful router must interpret hesitations, paraphrases, and corrections while keeping read-only observation separate from authority to act.

5.5 v0.5: conversation becomes measurable

v0.5 shifted from tuning individual replies to evaluating conversation as a system. A versioned corpus covered hundreds of isolated and multi-turn assertions; working state, response planning, bounded affect, dialogue stance, and prosody were separated so warmth could not change facts or tools. Voice identity and worker failure became explicit invariants. v0.5.9 then hardened filesystem paths, subprocess provenance, memory eligibility, logs, location helpers, restart behaviour, long conversations, injection handling, and voice identity. The outcome was not a finished personality. It was a disciplined baseline in which routing, affect, facts, and speech could fail independently and be tested independently.

5.6 v0.6: documents, projects, and a written constitution

v0.6 formalised the trust hierarchy—untrusted content, tool evidence, internal state, user direction, and policy—and encoded the rule that knowledge does not imply permission. Approved searchable roots, local indexing, passage-level citations, weak-match refusal, cross-document comparison, knowledge history, project scopes, continuity, protocol 1.0, exact model-call limits, and hardening followed across the family. The model treated document content as untrusted evidence rather than instruction. This family gave Iris its first durable answer to “where is that document?” without granting files the ability to control the assistant.

5.7 v0.7: operations, time, and the native client

v0.7 added an operations ledger, typed background jobs, supervised Core service, live Iris Desktop connection, document-presentation boundaries, active index refresh, read-only Calendar access, natural time questions, reminders, daily briefings, shadow proposals, confirmation, restraint, and stress testing. Later patches connected native Desktop voice, tightened permission sheets, repaired natural date and reminder wording, blocked invented actions, and prepared production installation. This era established the operational vocabulary that later releases kept: proposed, awaiting approval, executed once, verified, refused, failed, and recorded.

5.8 v0.8: credentials, Canvas, browser mediation, and unified voice

v0.8 introduced Keychain-backed credential foundations, authenticated Canvas reading, a fixed-origin browser observation bridge, receipt-bound Canvas navigation, individually confirmed submissions and messages, capability profiles, Desktop integration status, protocol-backed voice preferences, and typed document presentation controls. The v0.8.10 reliability line unified conversation and voice, stabilised capture teardown, focused document passages, shortened spoken clock replies, and added background Canvas observation. Browser Bridge remained separately versioned because Chrome service-worker lifecycle and signed-in site state are external constraints rather than properties of the Core alone.

5.9 v0.9: verified action and everyday personal operations

v0.9 began with a verified action lifecycle, Calendar and task mutations, sandboxed file operations with undo, broader Canvas awareness, bounded GitHub information, and email reading and drafts. Real use then exposed calendar-routing failures: “next week” was treated as one day, all-day spans were collapsed, and create requests fell into read routes. The v0.9.5 patch line repaired mutation ordering, voice availability, document-index truthfulness, natural ranges, verification, time windows, and multi-day presentation. v0.9.6–v0.9.8 added grounded daily planning, guarded Calendar management, hourly weather, practical rain phrasing, and deterministic barge-in.

The later 0.9 work made memories, projects, roots, jobs, connections, and other workspaces editable; added Outlook and Outlook Web boundaries; and built provider-neutral email intelligence with OAuth foundations, local indexing, classification, urgency, rules, attachments, drafts, actions, and explicit proactive-speech policies. v0.9.15–v0.9.19 connected those pieces into conversational inbox work, coverage reporting, cross-service handoffs, and scheduled briefings. v0.9.20–v0.9.23 added unified personal operations, conversational communication actions, grounded prioritisation, and a verified deadline watcher. Breadth increased, but every connection exposed new questions about freshness, partial coverage, and exact targets.

5.10 v0.10: ordinary language, continuity, and reviewed feedback

v0.10.0 turned familiar wording, typos, polite wrappers, indirect requests, and route collisions into a release-blocking corpus with evidence-policy assertions and a deterministic routing budget. v0.10.0.1 followed a concrete failure in which “show me that essay about Wave to Earth” returned an Apple order email; explicit current-turn document intent was made stronger than stale inbox context. v0.10.1 added selected-document continuity, local response feedback, and the Work/System navigation split. v0.10.2–v0.10.5 added sender-bound proactive speech authority, reviewed and sanitised Quality cases, current-list references, fresh Calendar corrections, native microphone noise handling, content-free voice timing, and installed release qualification. Feedback became input to a governed improvement process, not automatic model training.

5.11 v0.11: factual publication, latency, qualification, and recovery

v0.11.0–v0.11.3 introduced voice diagnostics, executable reviewed Quality cases, reversible presentation preferences, shorter grounded briefings, and natural follow-ups that did not restart with a greeting. v0.11.4 centralised factual publication checks; v0.11.5 clarified mailbox coverage and isolated browser sources; v0.11.6 grounded deadline follow-ups; v0.11.7 reduced scheduling, helper, network, and speech-chunk latency; v0.11.7.1 repaired equivalent Calendar phrasings; and v0.11.8 strengthened email topic ownership and cancellation. v0.11.9 compacted routing context and expanded realistic stress evaluation. The 0.11.9.1–0.11.9.5 patches then repaired spoken document correction, replacement-turn ownership, explicit email-list routing, compact email references, source-checked paraphrase, and selected-email elaboration.

The frozen v0.11.9.5 live qualification deliberately stopped feature work and tested everyday behaviour. Its failures led to v0.11.9.6 remediation of literal file reads, selected-item continuity, Canvas and Calendar windows, tasks and jobs, web receipts, and Desktop reset. v0.11.9.7 fixed “what about the next three weeks?” as a fresh 21-day Calendar read. v0.11.10 added the shared authority–coverage–freshness–health contract, provider and Canvas recovery, and a real read-only sync action. The v0.11.12 closure added exact owner-editable speech aliases, searchable conversation history, compact answer provenance, and integrity-checked backup and restore. Automated and installed-package gates passed; real acoustic and multi-day owner acceptance remain open rather than being inferred from source tests.

6. DESIGNING PRESENCE WITHOUT THEATRE

The Iris presence was one of the few concepts never removed. Layouts changed repeatedly—sidebar labels became compact, work and system sections were separated, dedicated voice buttons disappeared, and experimental centre-stage voice layouts were reverted—but the presence remained. Its value is functional: the user can click the same object to enter or exit voice, read state at a glance, and maintain spatial continuity while the conversation changes.

A key visual lesson was that dramatic motion can reduce perceived intelligence. When Iris moved left or right between modes, the interface felt unstable. When the speaking transcript repeated the same response above and below the presence, the design felt redundant. The final direction keeps Iris anchored, removes unnecessary dividing lines and duplicate copy, and lets state changes happen through colour, animation intensity, labels, and restrained spatial emphasis. Apple's Human Interface Guidelines similarly recommend using motion to support continuity and feedback rather than spectacle [17].

Design principle	Interface expression	Reason
Stable identity	Presence stays in one spatial home	Mode changes do not feel like a different product
Visible state	Standby, wake, listening, thinking, speaking	Users can infer what the system is doing
One conversation	Text remains available during voice	Typing and speech share context
Progressive evidence	Receipts expand on demand	Trust without constant visual noise
Accessible restraint	High contrast, labels, keyboard paths	Colour and animation are not the only signals

Table 2. Iris interface principles.

Accessibility remains part of the same argument. WCAG 2.2 treats contrast, focus visibility, target size, and non-colour cues as requirements for operable interfaces [16]. For Iris, accessibility also includes conversational pacing: voice state must be interruptible, long lists must be summarised, and important information must be both spoken and visible.

7. FAILURE AS THE DEVELOPMENT ENGINE

The most consequential releases were usually triggered by a mundane failure. “What do I have next week?” revealed brittle date routing. “Sure, what is that about?” exposed the loss of the selected email. “Could you provide more detail?” lost the current subject. “Show me that essay about Wave to Earth” crossed from document intent into email retrieval. “Do I need an umbrella in around an hour?” confused time with location. Interrupting speech worked only when the sentence began with “hold on.” These prompts were valuable precisely because they were not written like test commands.

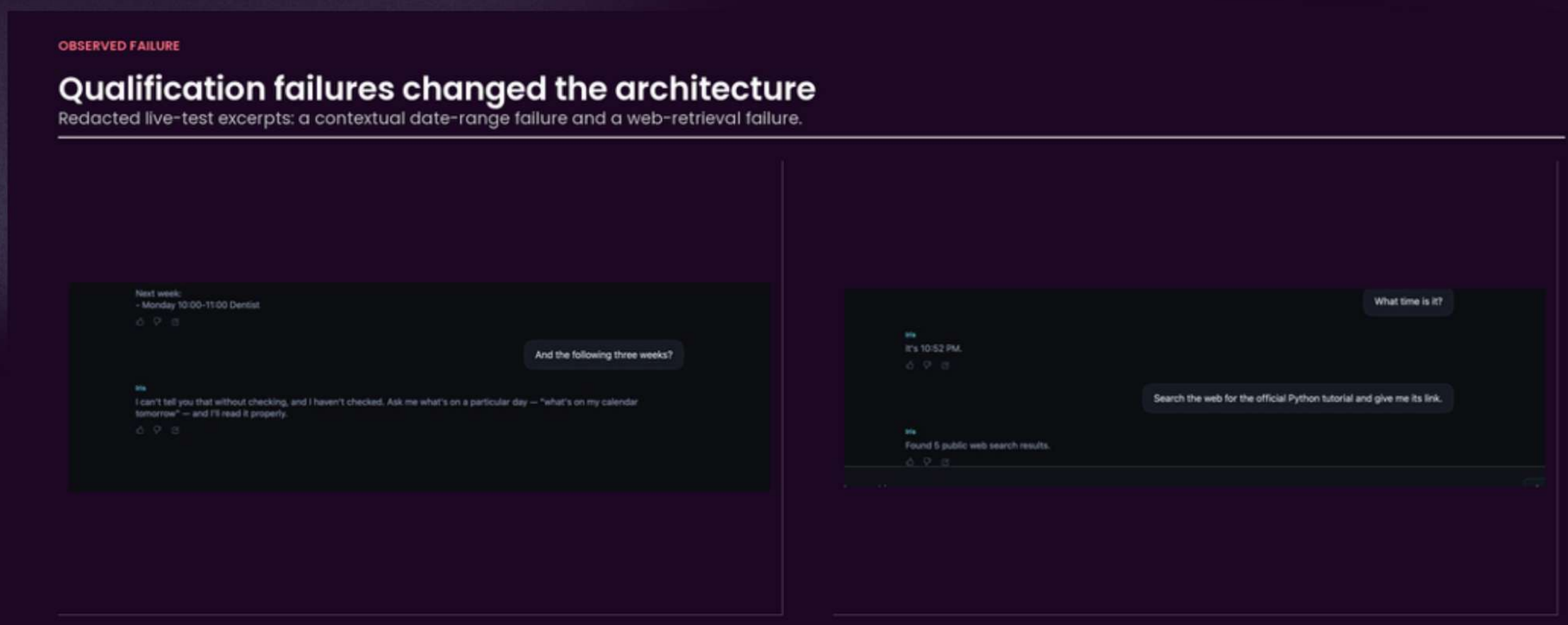


Figure 6. Redacted v0.11.9.5 live-test evidence. The crops retain system behaviour while removing unrelated personal content.

A second class of failures concerned fabrication. In a one-day briefing test, Iris confidently described appointments and a task that were not verified. The root issue was not merely tone; it was a missing integrity boundary between remembered language patterns and authoritative records. The qualification later found a high-severity memory problem in which negative preferences could be inverted. That affected trust across domains and stopped the affected test path until remediation.

7.1 Converting anecdotes into controls

The project avoided solving every incident with a larger prompt. Prompt text can guide behaviour, but recurring failures require state, types, and tests. A source receipt distinguishes retrieved material from conversational memory. A current-subject object preserves the selected email or document.

A coverage state prevents a six-message browser sample from becoming “your whole mailbox.” A mutation contract keeps drafting separate from sending. A Quality Corpus entry turns user feedback into a replayable candidate rather than an invisible preference.

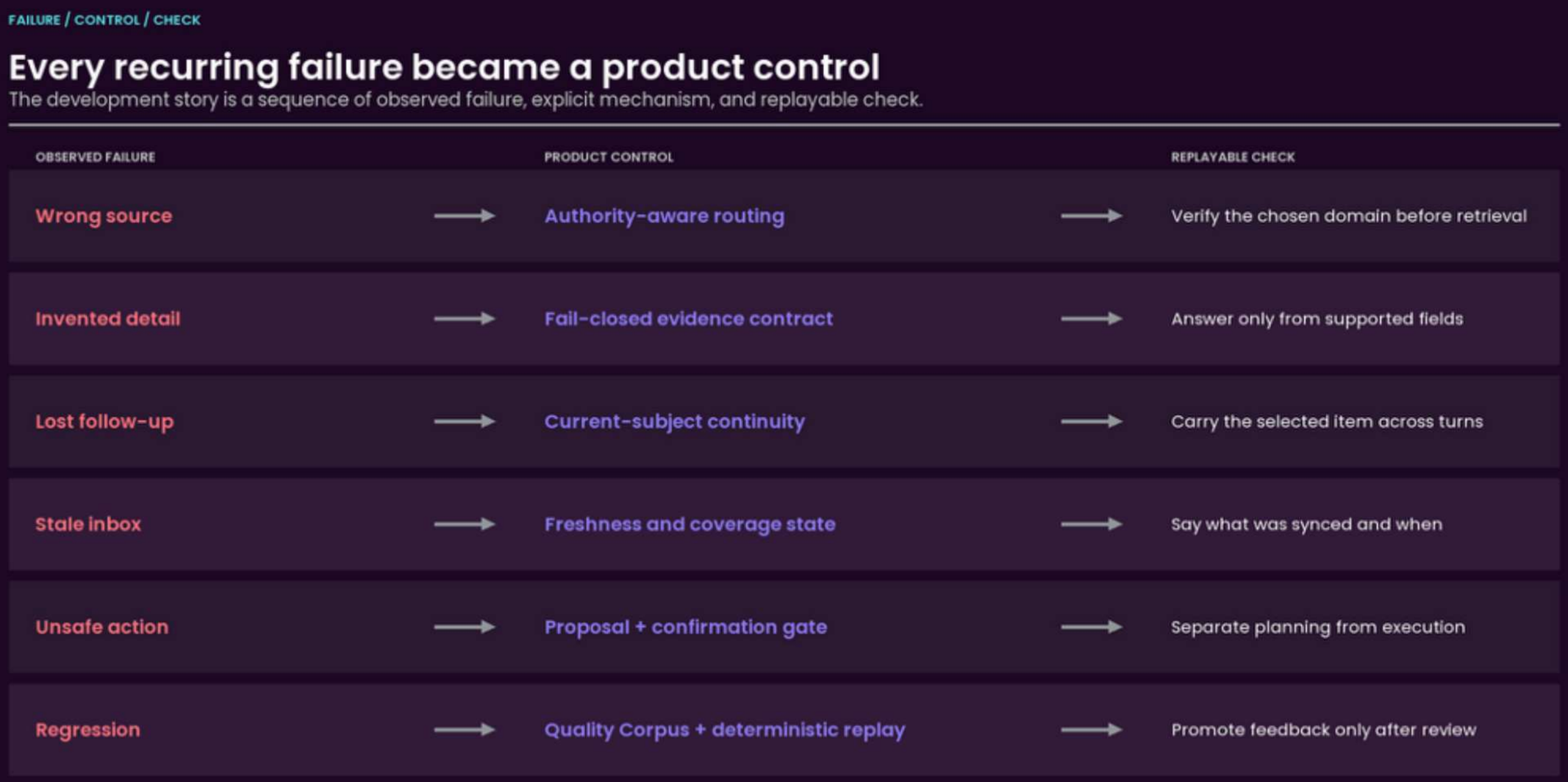


Figure 7. The project's recurring pattern: observable failures are converted into explicit controls and regression evidence.

7.2 Prompt injection as a boundary problem

Personal assistants ingest untrusted content from email, documents, web pages, and learning systems. Instructions inside those sources cannot be allowed to redefine the assistant's authority. OWASP classifies prompt injection as a primary risk for systems that combine language models with tools [18]. Iris's public architecture therefore treats external text as evidence to interpret, never as privileged instructions. That protection must continue to be tested across indirect injection, quoted messages, hidden web content, and generated attachments.

8. GROUNDING, CONTINUITY, AND NATURAL INTERACTION

8.1 Grounding is more than citation

A citation badge can prove where text came from, but it does not prove that the source was the correct authority, complete enough for the request, or still current. Iris's grounding model therefore has three layers. Retrieval grounding binds claims to records. Operational grounding binds the record to source state and permissions. Conversational grounding binds a follow-up to the correct current subject. The latter is easily overlooked: "what is that about?" has almost no semantic content without a selected item.

This is why natural follow-ups became a major 0.10 and 0.11 theme. Once the user and Iris establish an email, document, date range, or plan, later requests such as "more detail," "make it shorter," "where is it," or "what about the next three weeks?" should transform the current object rather than reclassify the entire conversation. ReAct-style systems demonstrate the value of interleaving reasoning and actions [5]; Iris adds an interaction requirement: the action history must remain legible enough to support deictic language.

8.2 Natural does not mean verbose

Early briefings were structured like system reports: headings for schedule, reminders, inbox, weather, coursework, tasks, and open questions, including sections that only said nothing was present. Spoken aloud, the result felt robotic. Later responses prioritised what changed, grouped related items, and omitted repeated greetings and capability disclaimers. A good briefing is not the same data rendered with friendlier adjectives. It is a different information hierarchy.

> **RESPONSE-QUALITY RULE**
>
> Lead with the decision-relevant fact; add evidence and caveats only where they change what the user should understand or do. Preserve an expandable receipt for inspection.

8.3 Corrections must update the subject

Speech recognition amplified continuity problems. "UChicago" was repeatedly misheard, and a correction such as "I meant UChicago" sometimes located the right file only for the next turn to open an unrelated PDF. The remedy requires speech aliasing and subject correction: normalise likely project terms, show the transcript, let the user correct it, and attach the correction to the active retrieval target. Even strong systems such as Whisper [7] need application-level adaptation for domain names and local acoustics.

9. EMAIL, CALENDAR, DOCUMENTS, AND DAILY WORK

9.1 Email as an interactive triage system

The email concept evolved from a recent-message list into an interactive triage experience. On a general check, Iris should report the number of new or recent messages, distinguish urgent from merely attention-worthy items, and summarise why they matter. The user can then ask about one sender, request more detail, draft a reply, or defer it. Raw subjects and snippets remain evidence; they should not be read aloud as the primary answer.

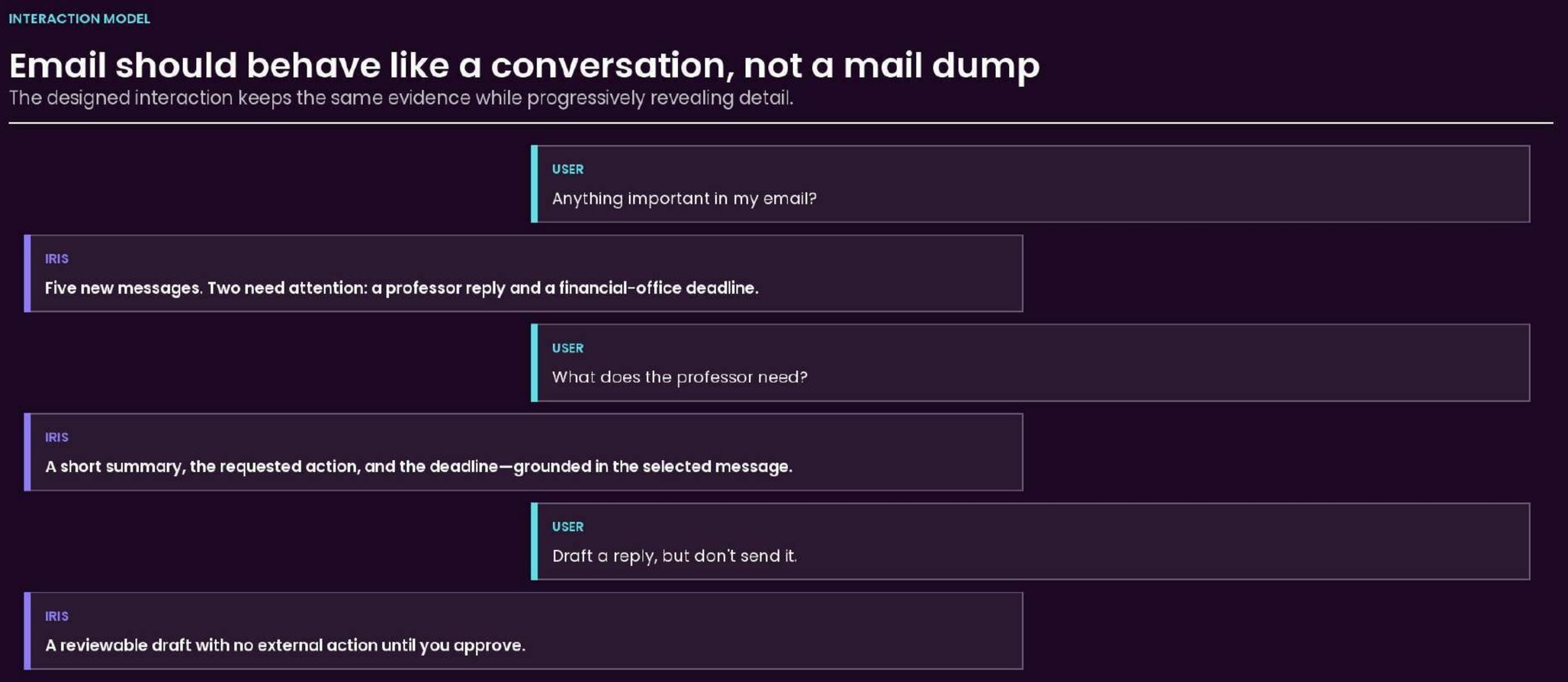


Figure 8. The intended progressive email interaction. External mutation remains behind explicit review.

Full provider access remains unfinished. The Browser Bridge can observe bounded Outlook Web content, but it cannot claim a whole mailbox. Provider-backed Outlook requires a correctly registered OAuth client and suitable Microsoft Graph permissions; delegated scopes are consented through OAuth, while broader application permissions may require administrator approval [9]–[11]. The user's school tenant did not permit the attempted application registration path, so the project retained browser observation as partial authority instead of bypassing the institution's controls.

9.2 Calendar as interval reasoning

Calendar reliability depends on interval semantics. “Next week,” “the following three weeks,” “this month,” and “in two weeks” must resolve relative to locale and current date. Multi-day all-day events are not single timestamps; an event from 1–3 September must be presented as a span or appear on all affected days. Follow-up expansion must trigger a new range query rather than reuse a single-day response. v0.11.9.7 specifically remediated this continuity path after a regression.

9.3 Documents as located evidence

Document retrieval has two user intents that should remain distinct: “show me the file” and “answer using the file.” The first requires a ranked file identity, location, and preview/open action. The second requires passage selection, page-level provenance, and a grounded synthesis. Current Iris previews include passage highlighting, source cards, preview, original-file opening, and “show details.” The system should ask when two documents match about equally rather than invent which one the user meant.

9.4 Canvas, weather, and the Browser Bridge

Canvas and web integrations illustrate why extension reliability is a systems concern. Chrome Manifest V3 service workers are intentionally ephemeral; they terminate when idle and must rebuild state from durable storage and events [12], [13]. Canvas APIs paginate responses and require callers to follow Link headers rather than assume the first page is complete [14]. These details can surface as conversational errors— missing assignments, stale data, or partial results—even when the language model is behaving correctly.

Weather added another lesson in intent decomposition. “Will I need an umbrella in around an hour?” contains a decision, a future time, an implied current location, and a request for hourly precipitation. Treating “around an hour” as a location produced an obviously wrong result. The corrected path extracts time and location separately, retrieves hourly conditions, and states the practical recommendation before supporting numbers. Unit preferences are stored in settings, while speech omits awkward suffixes such as “degrees C.”

10. VOICE, LATENCY, AND INTERRUPTION

Voice quality is a pipeline: wake detection, audio capture, endpointing, speech recognition, intent routing, tool execution, language generation, synthesis, audio startup, playback, and interruption. Faster hardware can shorten local inference and reduce memory pressure, but it cannot repair sequential network calls, redundant retrieval, cold process startup, poor endpointing, or a route that sends a simple clock question through a full model. Latency is therefore both a hardware and architecture problem.

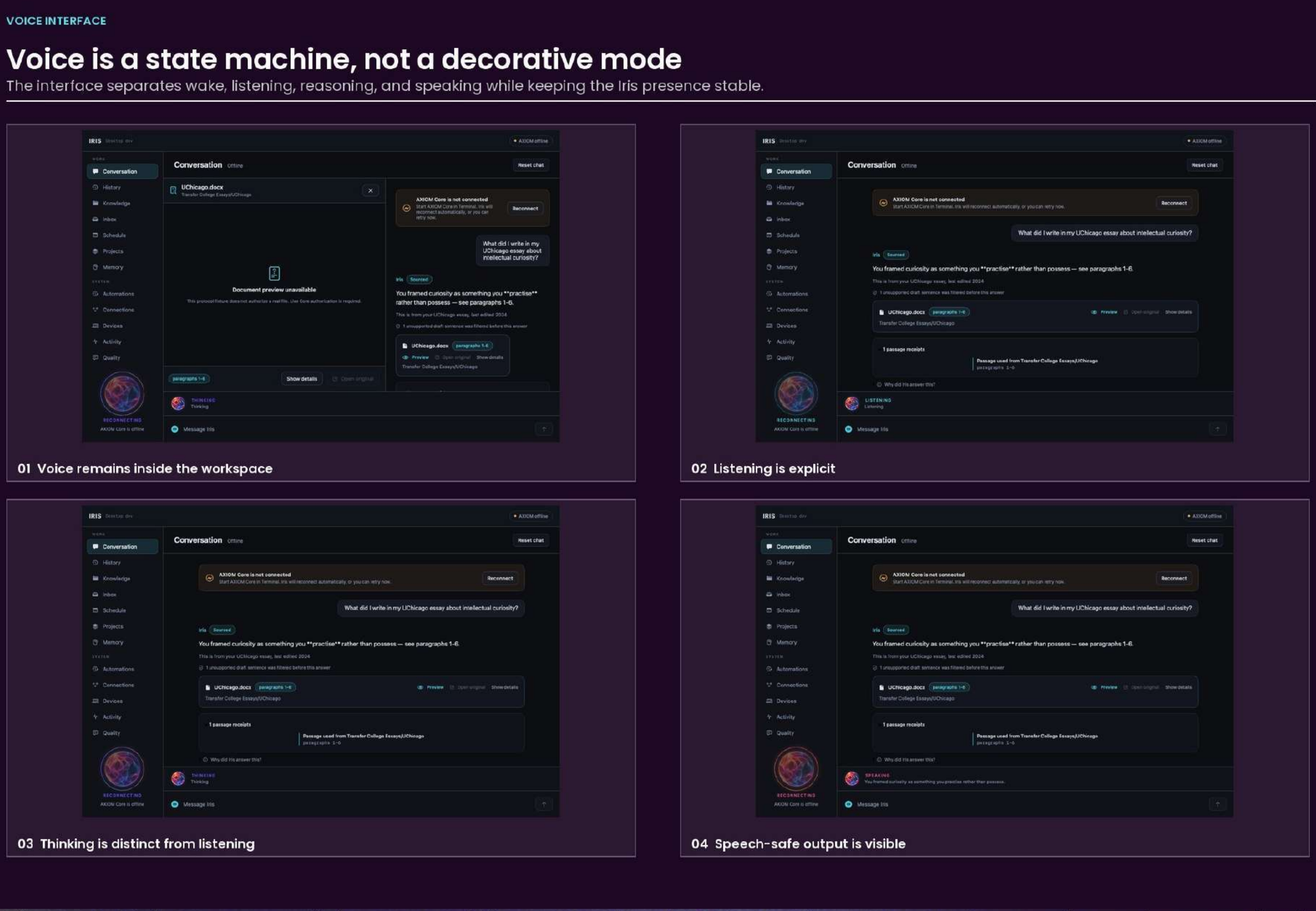


Figure 9. Voice uses explicit states while preserving the same Iris presence. Screenshots show synthetic development content.

10.1 Route-sensitive performance

The v0.11.9.5 live qualification measured Core completion time for 27 turns. The median was 2.850 seconds and the maximum was 26.212 seconds. A local clock fast path completed in 61 and 211 milliseconds, while wording that routed the same question through the model took 13.539 seconds. A short email summary took 14.250 seconds, a calendar-context failure took 23.729 seconds, and web final answers approached 26.2 seconds. The sample is heterogeneous, but the contrast is decisive: routing and orchestration dominated many delays.

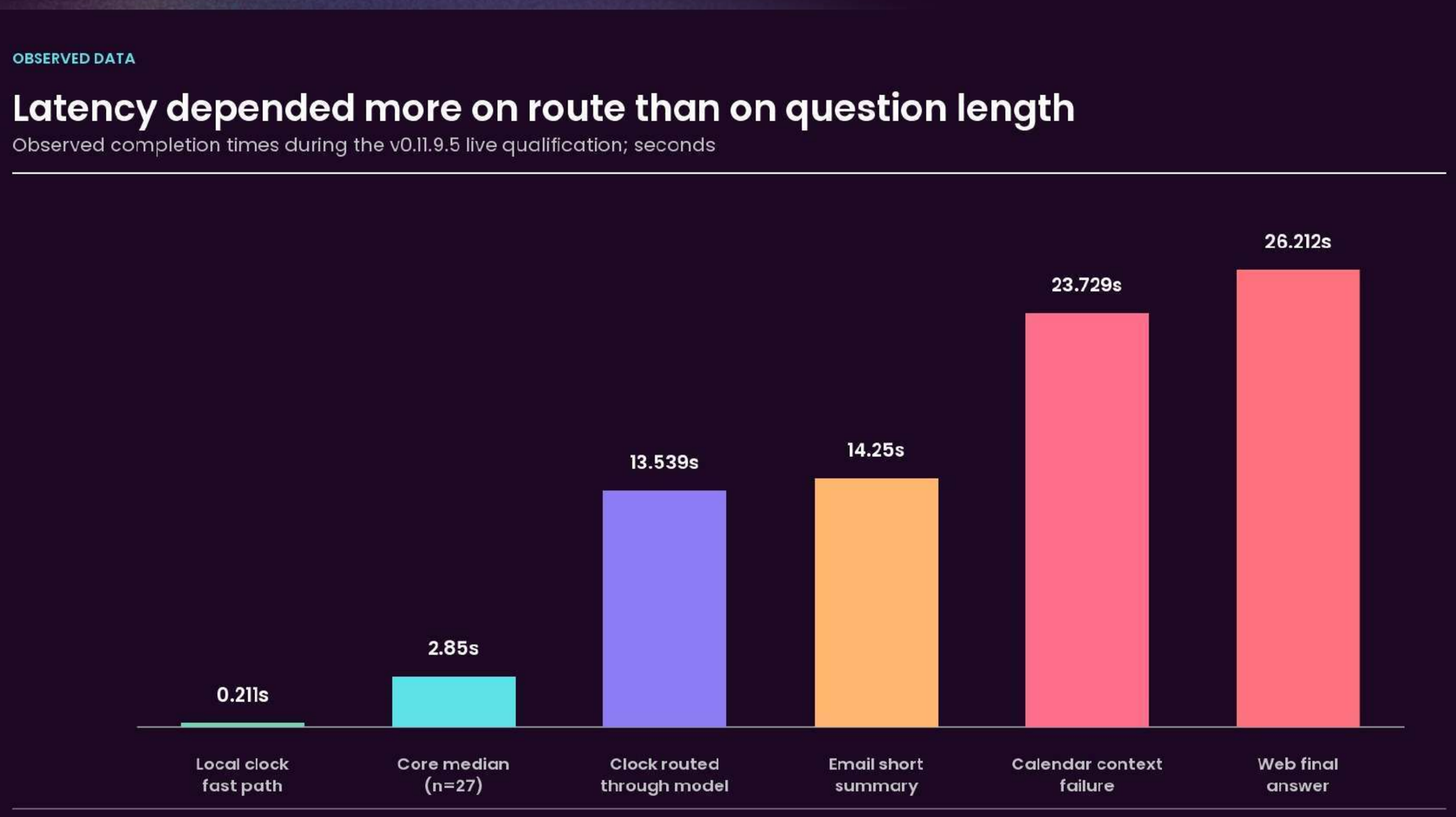


Figure 10. Selected completion timings from the v0.11.9.5 live qualification. Heterogeneous scenarios are shown to illustrate route sensitivity,

10.2 Cold speech and interruption

The user consistently observed that the first spoken response was much slower than subsequent turns. This points to cold initialisation: loading the speech engine or voice, preparing audio output, starting a model service, and populating caches. Later versions added warm-up and timing diagnostics, but a dispatched audio job is not the same as audible onset. A real microphone-and-speaker session remains necessary to qualify first-audio latency. Apple's Speech framework also makes authorisation, recognition tasks, audio sessions, and partial results explicit application responsibilities [8].

Interruption, or barge-in, was initially phrase-triggered. “Hold on” worked; arbitrary overlapping speech did not. Universal interruption requires concurrent capture while audio is playing, echo-aware voice-activity detection, a low-latency stop path, and careful handling of accidental noise. It should stop speech immediately, preserve the useful portion of the interrupted answer, transcribe the new request, and continue without a full reset. This remains a high-priority owner-assisted qualification item because replay files cannot reproduce loudspeaker echo and room acoustics.

11. QUALITY, QUALIFICATION, AND RECOVERY

11.1 The Quality Corpus is governed by feedback

User feedback becomes valuable when it can be inspected, edited, replayed, and promoted deliberately. Iris stores positive or negative judgement, an explanation, corrected wording where available, the original context, and review state. That record can generate a synthetic regression candidate, but it does not automatically train the active model or alter production behaviour. This separation reduces the risk of one frustrated click teaching a broad and unintended rule.

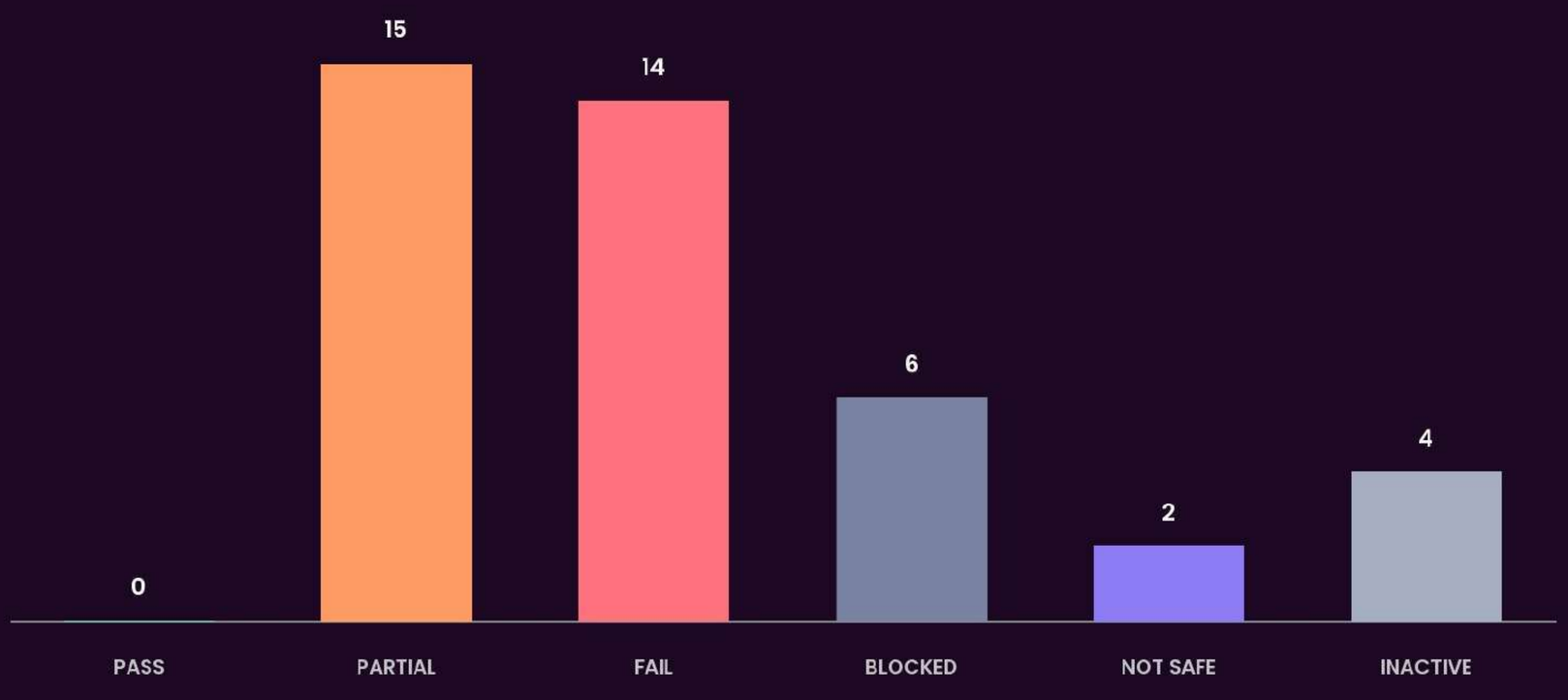
11.2 What the live qualification found

The v0.11.9.5 qualification covered forty-one capability families. None earned a full live PASS under the strict standard; fifteen were PARTIAL, fourteen FAIL, six BLOCKED, two NOT SAFELY TESTABLE, and four NOT CURRENTLY ACTIVE. The result does not mean every feature was broken. It means the build lacked repeated live evidence across the exact conditions required for a dependable claim. The strongest multi-source flow was the day briefing, yet it remained partial because connected coverage and naturalness were not consistently sufficient.

OBSERVED DATA

The live qualification exposed a reliability gap

Capability-family disposition in the v0.11.9.5 owner-like qualification (41 total families)



NOTE A PASS required repeated relevant live evidence. Automated coverage or one successful demonstration did not count as a live PASS.

Figure 11. Capability-family disposition from the frozen v0.11.9.5 live qualification.

11.3 Automated depth and its limits

Automated coverage expanded substantially. Selected checkpoints reported 1,541 Core tests at v0.9.23, 1,624 at v0.10.5, 1,646 at v0.11.3, 2,235 at v0.11.10, and 2,253 at v0.11.12. The current release also passed 59 Desktop tests, 11 Browser Bridge tests, six deterministic evaluation suites, and a packaged-file comparison in which 196 Core files matched deployment. These numbers show disciplined regression work. They do not convert blocked acoustic tests or unqualified live mutations into successes.

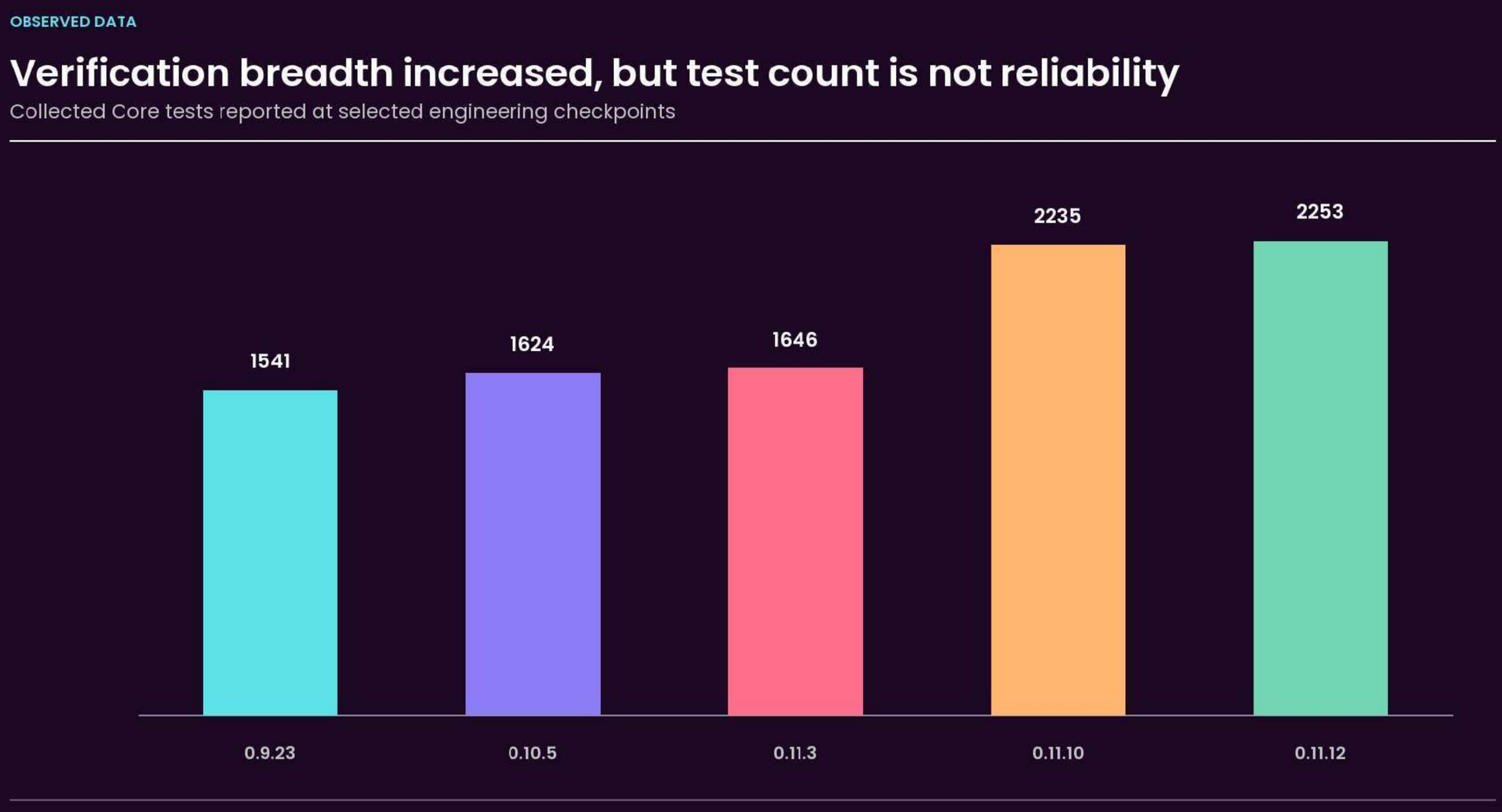


Figure 12. Selected Core test-suite sizes. Growth indicates broader regression coverage, not a probability of correctness.

11.4 Recovery is a user-facing capability

v0.11.12 added searchable, exportable, and deletable conversation history; “Why did Iris answer this?” evidence traces; and backup, list, and restore workflows with SHA-256 integrity checking and rollback. These features reflect a mature definition of reliability. Preventing failure is only one goal. The system must also help the owner understand, reverse, and recover from failures without silently corrupting memory or state. RELEASE GATE
A build is not dependable because it compiles, exports, or answers one demo prompt. The evidence must cover the installed artefact, connected sources, natural language, recovery, and real modality conditions.

RELEASE GATE

A build is not dependable because it compiles, exports, or answers one demo prompt. The evidence must cover the installed artefact, connected sources, natural language, recovery, and real-world modality conditions.

12. CURRENT STATE

What v0.11.12 Can and Cannot Claim

v0.11.12 closes the engineering work planned for the 0.11 family. Core and Desktop report the same version; protocol 1.23 is active; the installed package matches the source release record; and broad automated suites pass. The build includes exact speech aliases, bounded source-state reporting, contextual range continuation, selected-item follow-ups, reviewed quality cases, evidence traces, history management, and integrity-checked backup/restore.

The honest readiness statement is narrower than “Iris can help with anything.” It is a promising, increasingly accountable personal assistant for defined read-heavy workflows, with multiple connected and voice capabilities still dependent on environment, permissions, and live qualification. Provider-backed Outlook is not active without an approved OAuth client; Outlook Web remains partial authority. Devices are intentionally planned rather than implemented. Successful external mutations remain confirmation-gated and not comprehensively qualified against live providers. Real acoustic interruption, first-audio latency, and multi-day daily-use reliability remain open.

Capability area	Current evidence	Public claim
Conversation and continuity	Implemented; extensive regression; key follow-up paths remediated	Strong engineering evidence; longitudinal use still needed
Documents	Discovery, passage receipts, preview/open, ambiguity handling	Useful for indexed approved material
Calendar and planning	Natural ranges and interval fixes; read-heavy qualification	Capable reads; live writes remain gated
Email	Browser sample, triage, summaries, follow-up state	Partial mailbox authority without provider OAuth
Canvas and browser	Sync recovery and extension tests	Environment-dependent; not whole-account authority by default
Voice	Profiles, wake phrases, diagnostics, replay tests	Real acoustic barge-in and cold-start still unqualified
Quality and recovery	Editable feedback, evidence trace, history, backup/restore	Owner-controlled and inspectable
Connected devices	Planned placeholder	Not implemented

Table 3. Candid v0.11.12 capability statement.

12.1 Strongest capabilities

- Architecture-level provenance: the system increasingly records authority, coverage, and freshness. and the evidence path behind an answer
- Document and current-subject workflows: sources can be previewed, opened, and discussed across follow-ups with ambiguity handling
- Owner control: memory, feedback, history, permissions, and recovery are visible rather than hidden inside model behaviour
- regression discipline: previously reported natural prompts are retained as tests instead of being treated as one-off demos

12.2 Weaker or least-qualified capabilities

- true whole-mailbox Outlook access under an institution-controlled tenant
- Universal spoken interruption and measured first-audio latency in the owner's acoustic environment
- Consistent low latency for tool-heavy and web-research turns
- Comprehensive live external mutations and cross-provider recovery
- Longitudinal evidence that the installed build remains dependable across days, restarts, new mail, and changing schedules

13. ROADMAP TO V1.0 AND BEYOND

The roadmap deliberately resists treating v1.0 as the next number after 0.11. Each remaining family has a different burden of proof and depends on the earlier evidence architecture. Research must preserve sources and contradictions. Finance must reproduce calculations and periods. Composed workflows must preserve authority through several steps. New compute must win measured comparisons rather than conceal software defects. Hardening must make setup, privacy, failure, and recovery manageable by a non-developer. Only then does a frozen candidate enter an extended daily-use qualification.

Figure 13. Planned release-family sequence to v1.0. The final gate is a frozen build followed by fourteen days of qualification.

13.1 v0.12.x: evidence-led general and academic research

v0.12.0 establishes a research workspace with a typed question, scope, budget, stop conditions, query history, source acquisition beyond snippets, duplicate detection, source-quality labels, claim-level provenance, and resumable jobs. v0.12.1 adds synthesis that separates facts, source claims, calculations, assumptions, inference, and uncertainty while actively searching for contradiction and missing evidence. v0.12.2 applies the contract to academic work: primary papers, methods, samples, limitations, citation traversal, bibliographies, and quote verification. v0.12.3 adds personal intellectual continuity across essays, notes, and versions without treating a past argument as a current belief. v0.12.4 is the held-out qualification release.

The family earns closure only when a reviewer can trace each material claim to an inspected source, reproduce any calculation, and see unresolved gaps. Search is therefore an acquisition mechanism, not the answer itself. Paywalls, unavailable papers, prompt injection, conflicting sources, and provider outages are first-class test conditions.

13.2 v0.13.x: checked financial analysis and modelling

v0.13.0 normalises company, security, reporting period, fiscal calendar, currency, units, restatements, and source date across filings and earnings material. v0.13.1 reconstructs statements, ratios, margins, working capital, dilution, segments, concentration, and accounting-quality flags with deterministic tie-outs. v0.13.2 adds reproducible discounted-cash-flow, comparable-company, historical-multiple, bear/base/bull, and sensitivity models with editable assumptions and visible formulas. v0.13.3 supports investor dialogue, catalysts, risks, thesis breakers, and updates that preserve prior assumptions. v0.13.4 independently qualifies extraction and arithmetic.

This family is intentionally bounded. It does not include brokerage access, autonomous trading, personalised investment instructions, or an attempt to reproduce every institutional data product. Important numbers require lineage; unresolved reconciliation blocks confident publication; the model computes mathematics rather than improvising it.

13.3 v0.14.x: composable personal workflows

v0.14 moves from isolated tools to finite, reviewable workflows. A typed planner describes inputs, dependencies, budgets, permissions, outputs, and stop conditions. A durable state machine supports queued, running, awaiting approval, paused, completed, failed, cancelled, and superseded work with idempotent restart recovery. Daily operations then combine verified Calendar, deadlines, tasks, projects, travel time, weather, and explicit preferences for requests such as “handle my morning” or “I have ninety minutes—what should I do?” Cross-service actions remain exact-target proposals with final preview, confirmation, read-after-write verification, and repair guidance.

13.4 v0.15.x: compute, Brain, and voice architecture

v0.15 uses the planned 64 GB Mac mini as a measured migration rather than a promise that hardware solves design. v0.15.0 rehearses encrypted backup, restore, rollback, and reauthorisation while retaining the M2 fallback. v0.15.1 compares viable local models, quantisations, Ollama, llama.cpp, and MLX on identical held-out Iris workloads. v0.15.2 introduces capability-aware routing and resource management for fast conversation, deep research, code, embeddings, ASR, and speech while preserving one Iris identity and policy layer. v0.15.3 evaluates higher-quality streaming speech, recognition, prosody, names, dates, units, and interruption. v0.15.4 stresses concurrent work, sleep/wake, provider outages, thermals, disk, memory, and restart.

13.5 v0.16.x: security, performance, and product hardening

v0.16 threat-models every authority boundary, including documents, browser content, providers, jobs, actions, skills, and local IPC. It adds adversarial attachment and page tests, confused-deputy and duplicate-action checks, expired approvals, revoked-source behaviour, signed migrations, automatic pre- update backup, rollback, support diagnostics that exclude private content, complete onboarding and account repair, retention and deletion controls, accessibility, offline operation, storage and log management, and explicit performance and resource budgets. A non-developer should be able to install, connect, diagnose, recover, and understand the supported capability contract without editing configuration files.

13.6 v1.0 release candidate and qualification

Post-v1 work preserves the larger concept without overloading the first dependable release. Separate programmes would cover authenticated iPhone, iPad, and room endpoints; speaker-aware identity and guest privacy; a restrained smart-mirror and ambient display; opt-in screen or camera perception with short retention; broader application control; calls and general messaging; CAD and physical-design assistance; and more capable engineering workflows. Each adds new privacy and authority questions and therefore needs its own threat model, consent model, capability manifest, and live qualification.

13.7 Beyond v1.0: ambient, multi-device, and expanded agency

Post-v1 work preserves the larger concept without overloading the first dependable release. Separate programmes would cover authenticated iPhone, iPad, and room endpoints; speaker-aware identity and guest privacy; a restrained smart-mirror and ambient display; opt-in screen or camera perception with short retention; broader application control; calls and general messaging; CAD and physical-design assistance; and more capable engineering workflows. Each adds new privacy and authority questions and therefore needs its own threat model, consent model, capability manifest, and live qualification.

Self-improvement remains governed rather than autonomous. Feedback may become a reviewed Quality candidate; a privacy-safe regression may justify an isolated patch; gates and installed replay may support owner promotion. Raw ratings, silence, or performance metrics do not grant Iris permission to retrain, edit, deploy, purchase, trade, or expand her own authority. The long-term destination is broad and ambient assistance, not invisible control.

14. LESSONS FOR BUILDING DEPENDABLE PERSONAL AI

14.1 The model is only one source of latency and error

The same installed model answered a clock question in milliseconds or more than thirteen seconds, depending on the route. Similarly, a correct retrieval could still become a poor answer through stale coverage, weak ranking, or lost subject state. Architecture and interaction design therefore matter as much as model selection. More memory and a faster Mac can improve local inference, but efficient routing, warm processes, parallel retrieval, caching, streaming, and early cancellation determine whether those gains are visible.

14.2 Disclaimers are not a substitute for source state

Repeating “I can only see the browser-synced sample” on every email answer made Iris feel robotic without improving capability. The better design records the limitation once, incorporates it into wording only when it changes the answer, and offers a visible receipt for inspection. Trust comes from accurate behaviour plus accessible explanation, not constant self-exoneration.

14.3 Personalisation must be reversible

A preference, memory, or feedback item is not harmless just because it is local. The qualification's negative-preference inversion showed that incorrect personalisation can contaminate every later turn. Personal state needs provenance, confidence, editing, deletion, backup, and rollback. Local-first ownership increases the user's control; it does not eliminate the need for integrity.

14.4 Natural interaction is a data-flow requirement

Naturalness is often treated as tone. Iris showed that it is mostly continuity. “That,” “more,” “the next three weeks,” and “make it shorter” depend on a stable representation of the current subject and operation. Without that representation, stylistic polish cannot rescue the exchange.

14.5 Public storytelling improves when limitations are explicit

The project becomes more credible when it shows failed screens, blocked provider access, and unqualified voice claims alongside polished UI. The narrative is not that Iris steadily accumulated features. Instead, it shows that the definition of “working” became stricter. This approach matches the broader movement toward documented risk, transparency, and evaluation in generative AI systems [1], [2].

15. LIMITATIONS, ETHICS, AND CONCLUSION

15.1 Limitations of this case study

This paper is based on one owner's system, local environment, and development record. It is not a controlled comparison against commercial assistants, a statistically powered user study, or an independent security audit. Some metrics come from different scenarios and should not be directly compared. Live qualification was deepest at v0.11.9.5; later releases have stronger automated and package verification but do not yet have an equivalent fourteen-day owner-use record. Private evidence has been redacted or summarised, which limits external reproducibility.

15.2 Ethical boundary

A comprehensive assistant should not become an invisible surveillance system. Iris's future design should preserve explicit source connections, least privilege, per-action confirmation where consequences are meaningful, deletion and export rights, and a visible explanation for proactive alerts. Urgent email may justify Iris speaking first; the owner must still be able to define urgency, quiet hours, eligible senders, and whether any content is spoken aloud.

15.3 Conclusion

Iris began as an interface for an imagined kind of relationship with computing: present, conversational, and capable of helping across domains. The engineering journey showed that the central challenge is not making the assistant appear intelligent. It is making intelligence accountable to evidence, permissions, context, and recovery. Each ordinary failure—an email quoted instead of summarised, an all-day event reduced to one date, a document confused with a message, a follow-up that lost its subject, a voice turn that could not be interrupted—revealed a missing piece of that accountability.

v0.11.12 is therefore best understood as a foundation rather than a finish line. It closes a major reliability family with broader regression, stronger source contracts, explainable answers, and recoverable local state. The remaining path to v1.0 is intentionally demanding: research and financial reasoning must become evidence-backed; composition must preserve provenance; voice and compute must be measured in real conditions; security and provider integrations must withstand failure; and a frozen build must survive prolonged daily use. If those gates are met, Iris will not be a system that can do anything literally. It will be something more useful: a personal assistant whose capabilities are broad, legible, and dependable enough to trust.

FINAL POSITION

The project's strongest idea is not the luminous presence or the number of tools. It is the decision to make uncertainty, source state, consent, feedback, and recovery part of the product itself.

REFERENCES

External sources

[1] National Institute of Standards and Technology, "AI Risk Management Framework (AI RMF 1.0)," Jan. 2023. NIST AI RMF

[2] National Institute of Standards and Technology, "Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile," NIST AI 600-1, July 2024. NIST GenAI Profile PDF

[3] S. Amershi et al., "Guidelines for Human-AI Interaction," in Proc. CHI 2019, doi: 10.1145/3290605.3300233. Microsoft Research paper

[4] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," arXiv:2005.11401, 2020. arXiv 2005.11401

[5] S. Yao et al., "ReAct: Synergizing Reasoning and Acting in Language Models," arXiv:2210.03629, 2022. arXiv 2210.03629

[6] M. Kleppmann et al., "Local-first software: You own your data, in spite of the cloud," Onward! 2019. Local-first paper

[7] A. Radford et al., "Robust Speech Recognition via Large-Scale Weak Supervision," in Proc. ICML 2023. PMLR Whisper paper

[8] Apple Developer Documentation, "Speech." Apple Speech documentation

[9] Microsoft Learn, "Permissions and consent in the Microsoft identity platform." Microsoft permissions guide

[10] Microsoft Learn, "OAuth 2.0 authorization code flow." Microsoft OAuth guide

[11] Microsoft Learn, "Microsoft Graph permissions reference." Microsoft Graph permissions

[12] Chrome for Developers, "The extension service worker lifecycle." Chrome worker lifecycle

[13] Chrome for Developers, "Manifest V3." Chrome Manifest V3

[14] Instructure Canvas LMS REST API, "Pagination." Canvas pagination

[15] Instructure Canvas LMS REST API, "Assignments." Canvas assignments API

[16] World Wide Web Consortium, "Web Content Accessibility Guidelines (WCAG) 2.2." W3C WCAG 2.2

[17] Apple, "Human Interface Guidelines: Motion." Apple motion guidance

[18] OWASP GenAI Security Project, "LLM01: Prompt Injection." OWASP prompt injection

[19] NIST, "Zero Trust Architecture," NIST SP 800-207, Aug. 2020. NIST zero-trust PDF

[20] T. Schick et al., "Toolformer: Language Models Can Teach Themselves to Use Tools," arXiv:2302.04761, 2023. arXiv 2302.04761

REFERENCES

Project records

[PR1] AXIOM source history, tagged releases, CHANGELOG, roadmap, Constitution and architecture records from project initialisation through v0.11.12.

[PR2] AXIOM Decision Log and Concept Continuity record, 30 Aug.–4 Sep. 2026.

[PR3] AXIOM v0.4.9 conversation baseline and v0.5 evaluation, working-state, response-planning, affect, dialogue, voice and hardening reports

[PR4] AXIOM v0.6 Constitution, indexing, retrieval, grounding, comparison, knowledge, continuity, projects and hardening reports.

[PR5] AXIOM v0.7 ledger, service, jobs, Calendar, time, reminders, briefing, shadow-action, restraint, stress and production-readiness reports. [PR6] AXIOM v0.8 credential, Canvas, browser, provenance, voice and reliability reports.

[PR7] AXIOM v0.9.23 Acting-Family release report and verification ledger.

[PR8] AXIOM v0.10 Family report and cross-domain retrieval remediation notes.

[PR9] AXIOM v0.11 plan and v0.11.0–v0.11.3 release report.

[PR10] AXIOM/Iris Live Qualification Report, Capability Matrix, Transcript, Findings, Quality Corpus Candidates, and Remediation Plan for v.11.9.5, 31 Aug. 2026.

[PR11] AXIOM/Iris v0.11.9.6 remediation and v0.11.9.7 Calendar range reports.

[PR12] AXIOM v0.11.10 source-state and provider-recovery release records.

[PR13] AXIOM v0.11.12 Reliability Closure Report, including installed-package verification.

[PR14] AXIOM/Iris v1.0 plan and release-family gate definitions.

[PR15] Iris Desktop source history and public-safe preview image set, current through v0.11.12.

[PR16] Iris Browser source history, capability records and background-synchronisation tests.

[PR17] AXIOM/Iris public showcase, release summary, development story, migration plan, roadmap and nineteen-scenario vision record.