SALESFUSION

Ask Once, Analyze Many

Technical architecture and implementation report

PUBLIC RELEASE · VERSION 1.0.0 · 2026-07-17

Ask Once, Analyze Many

Publication metadata

Field	Value
Author	SalesFusion Research
Publication date	2026-07-17
Version	1.0.0
Canonical slug	ask-once-analyze-many
Canonical path	/technical-papers/ask-once-analyze-many
Publication class	Technical architecture and implementation report
Approval state	Founder-approved public release

Revenue Cortex connection

Revenue Cortex should not ask a founder to repeat the same company facts for every asset, agent, audit, or campaign. A canonical-state model lets approved inputs about the product, buyer, offer, objections, proof, and constraints feed many analyses while keeping each downstream use reviewable and bounded.

Evidence-status box

> This paper describes an implemented SalesFusion interaction-architecture redesign and states its evidence limits. SalesFusion removed 24 duplicate designed tasks across the first eight modules of a Revenue OS design artifact and corrected the portal interaction model around canonical state and delta-only review. That count is a design artifact count, not a measured client outcome, time-savings result, or usability result. The broader claim remains to be evaluated: canonical state and delta-only review are intended to remove repeated client input while preserving distinct analytical and authority boundaries.

Abstract

Multi-stage agentic workflows often ask humans the same question repeatedly because each module treats its own form, prompt, or checklist as the source of truth. Repetition creates fatigue, inconsistency, and review noise, but removing all review is unsafe. We define "Ask Once, Analyze Many" as an interaction architecture in which human input becomes canonical state, later stages request only deltas or unresolved fields, and analysis remains separate from authority. A client or operator answers a question once when the answer is stable,

source-backed, and properly scoped. Analytical agents can reuse that canonical state for multiple tasks, but they cannot convert analysis into approval or mutate consequential systems without the required authority. The paper specifies canonical state objects, delta review, boundary rules, state transitions, evaluation design, and failure modes. It is an implementation report about interaction design, not evidence of client productivity improvement.

The problem

Repeated questioning is common in consulting portals, onboarding workflows, RevOps audits, and AI-assisted operating systems. A module asks for objectives. A later module asks for the same objective in a different phrasing. A third workflow asks for the target segment again because it does not trust the prior answer. The person answering begins to copy, abbreviate, or contradict prior responses. The system then has more text but less reliable state.

The opposite failure is also common. A system asks once, stores a vague answer, and reuses it everywhere without checking whether the later use has a different audience, authority requirement, effective time, or restriction. That creates hidden overreach. Input reuse is only safe when the system knows what the answer means, where it came from, when it applies, and what it may authorize.

The required architecture is therefore not "ask fewer questions." It is canonical state with deltas, evidence, review states, and authority boundaries.

Contributions

This paper contributes four implementation concepts.

First, it defines canonical state as the reusable record created from human input, source evidence, accepted corrections, and versioned review.

Second, it defines delta-only review: later stages present what changed, what is missing, what conflicts, and what requires authority, rather than asking for everything again.

Third, it separates analytical reuse from approval. Reusing an answer for analysis does not authorize an external action or policy change.

Fourth, it states an evidence-limited implementation result: 24 duplicate designed tasks were retired from a design artifact. The paper does not claim measured effort reduction.

Core definitions

Canonical state is the structured, versioned record that later workflows read. It contains values, sources, owners, effective time, review status, restrictions, and history.

Question instance is a human-facing request for information. It maps to one or more canonical fields.

Delta is a change, missing field, conflict, stale field, restriction issue, or authority issue relative to the last accepted canonical state.

Delta-only review is a review mode in which the user sees only the fields requiring attention and the reason each field requires attention.

Analytical boundary is the limit between using state to generate analysis and treating that analysis as accepted company fact.

Authority boundary is the limit between analysis or recommendation and consequential action.

Review state is the status of a canonical field: draft, proposed, active, stale, disputed, superseded, restricted, or unknown.

Interaction architecture

The architecture begins with a field map. Each question must declare which canonical field it affects, what source or actor can answer it, whether the value is stable or time-bound, and what later workflows may use it for.

```
{
  "field_id": "company.current_priority_constraint",
  "question_text": "What constraint currently limits the operating motion?",
  "required_source": "owner_review_or_source_record",
  "effective_time_required": true,
  "allowed_uses": ["analysis", "internal_recommendation"],
  "disallowed_uses_without_approval": ["external_action", "policy_change"],
  "review_cadence": "monthly"
}
```

The canonical field record:

```
{
  "field_id": "company.current_priority_constraint",
  "value": "operator approval required for external actions",
  "state": "active",
  "source_refs": ["src_authority_matrix_v1"],
  "owner_role": "business_owner",
  "effective_from": "2026-07-01",
  "effective_to": null,
  "last_reviewed_at": "2026-07-15",
  "review_due_at": "2026-08-15",
  "restriction_refs": [],
  "version": 4
}
```

Later modules do not ask the original question again unless a rule requires it. They request a state snapshot. The response can say:

State	User-facing behavior
active and fresh	do not ask again; show in review context if needed
missing	ask once and bind to canonical field
stale	ask for confirmation or new source
disputed	show conflict and request resolution
restricted	hide or request authorized reviewer
changed	show delta and ask for acceptance
analysis-only	allow reuse for reasoning, not action

This creates a practical difference between repeated input and repeated confirmation. The system should not re-collect stable facts. It should still surface meaningful deltas and approvals.

Analytical and authority boundaries

The design separates four states that are often collapsed:

Boundary	Allowed movement	Not allowed
input to canonical state	answer becomes proposed or active field after validation	raw text silently becomes universal truth
canonical state to analysis	agents reuse source-backed fields for analysis	agents ignore restrictions or freshness
analysis to recommendation	system proposes options with evidence	recommendation becomes approval
approval to execution	approved action can be executed by separate step	approval event directly mutates external system

This matters because asking once can otherwise become unsafe. A user may answer a strategy question for internal analysis. That answer should not automatically authorize external contact, commercial commitment, publication, or production mutation. Consequence class and policy still apply.

The architecture also guards against generated analysis contaminating canonical state. An agent may infer a likely missing field, but the inferred value remains proposed or inferred until accepted through the right review path.

Delta-only review protocol

A delta review packet contains:

Field	Meaning
changed fields	accepted value, proposed value, source, and diff
missing fields	required fields absent for this workflow
stale fields	fields past review date or effective interval
conflicts	contradictory active or proposed values
restricted fields	values unavailable for current actor or output
authority requests	approvals needed for the next consequential step
no-change digest	integrity summary of reused state

The review algorithm:

```
snapshot = canonical_state.snapshot(objective, workflow_version, actor)
delta = compare(snapshot, workflow_requirements)

if delta.has_missing_required_fields:
    ask_questions(delta.missing)
if delta.has_stale_fields:
    request_confirmation_or_source(delta.stale)
if delta.has_conflicts:
    request_resolution(delta.conflicts)
if delta.has_authority_requests:
    create_approval_packet(delta.authority)
if delta.is_empty:
    proceed_to_analysis_with_digest(snapshot.digest)
```

The user should be able to inspect reused state without being forced to retype it. The system should also be able to explain why a question is being asked again. Acceptable reasons include staleness, conflict, changed use, changed authority, restriction, or missing source.

Implementation

SalesFusion implemented a redesign that retired 24 duplicate designed tasks across the first eight modules of a Revenue OS artifact. The redesign changed the interaction model from repeated module-local questions to canonical state and delta review. The count is only evidence that designed duplication was removed from the artifact. It does not prove that users saved time, made fewer errors, or completed work faster.

An implementation requires:

Component	Role
canonical field registry	maps questions to reusable state
source and assertion ledger	preserves provenance and effective time
workflow requirement schema	declares needed fields and authority
delta engine	computes missing, changed, stale, disputed, restricted fields
review UI	presents deltas and accepts corrections
approval service	handles consequential authority separately
audit events	records question, answer, reuse, correction, approval

Events include field.proposed, field.activated, field.corrected, field.marked_stale, field.superseded, review.delta_created, review.accepted, review.rejected, approval.requested, and decision.recommended.

The design uses the Company Memory principle that correction creates a new record or version; it does not erase prior state. Obsidian or a portal can be the human surface, but canonical state lives in structured records.

Evaluation

The correct evaluation is not an artifact count alone. A before/after study should measure human task success, time on task, repeated-question exposure, correction rate, contradiction rate, decision quality judged by reviewers, authority comprehension, and support requests. It should also record null and negative results.

Scenario tests should include stable fields reused across modules, fields that become stale, conflicting answers from two reviewers, a restricted answer requested for public output, an answer valid for analysis but not external action, and a generated inference attempting to become canonical state.

Acceptance criteria:

Criterion	Required behavior
no duplicate collection	active fresh field is not re-asked as blank input
explain repeat	repeated question has a visible reason
delta accuracy	changed, missing, stale, disputed, restricted fields are correctly labeled
boundary preservation	analysis reuse does not create approval
correction integrity	prior value remains auditable
authority separation	consequential action still requires policy decision

This evaluation would support usability and interaction claims only if method, sample, baseline, and results are published together.

Safety, risks, and failure modes

The main risk is over-reuse. A field that was correct for one workflow may be wrong for another because the effective time, audience, authority, or restriction differs. Another risk is under-review: the system hides a reused field that should have been confirmed. The opposite risk is false duplication, where a system asks again because it cannot map question variants to the same canonical field.

Controls include field-level effective time, allowed-use metadata, restriction checks, review cadence, authority classification, delta reasons, and audit events. NIST AI RMF supports the general need to manage AI-system risk; W3C PROV-O supports provenance vocabulary for state and source relationships. These references do not establish usability outcomes.

Limitations

Canonical state requires careful field design. If fields are too broad, reuse becomes unsafe. If fields are too narrow, users face unnecessary deltas. Delta-only review can also hide context if the interface does not let reviewers inspect the full state when needed.

The 24-task correction is an implementation artifact count. It is useful evidence that the design was changed, but it is not evidence that client effort decreased. A proper study may find smaller benefits, no benefits, or tradeoffs between reduced repetition and increased need to understand canonical state.

Practical implications

Teams building multi-stage agent workflows should create a canonical field registry before expanding modules. Every repeated question should be classified as one of four cases: same canonical field, genuinely new field, stale confirmation, or authority-specific review. This classification turns "ask less" from a preference into a rule.

Operators should monitor repeated-question reports and correction logs. If users repeatedly correct a reused value, the field definition, review cadence, or source authority may be wrong. If users repeatedly approve actions without reading deltas, the authority UI needs redesign.

References used

This paper uses W3C PROV-O for provenance vocabulary and NIST AI Risk Management Framework for risk framing. These references do not validate SalesFusion-specific usability or business outcomes.

Conclusion

Ask Once, Analyze Many is a canonical-state architecture, not a slogan. The system should collect stable information once, preserve its evidence and limits, and reuse it across analytical workflows through delta-only

review. SalesFusion implemented a design correction that retired 24 duplicate tasks from an artifact, but the practical effects remain unevaluated. The next step is a before/after usability and decision-quality study with visible method and limitations.

Changelog

- 2026-07-17 — Version 1.0.0: founder-approved first public release; publication metadata, evidence-status box, limitations, and verified references retained.