TOUCH GRASS CONSULTING

# Agentic Experience 2026

## Principles & Framework

### A Draft Framework for Autonomous, Intent-Based Interfaces

We're past the prototype phase. Agentic systems are shipping — approving expenses, scheduling meetings, generating code, managing workflows. But the design patterns are still borrowed from chatbots and dashboards.

We need a native language for interfaces where machines act on our behalf.

**DRAFT V0.2** | 2026-02-04 | Active Working Document

CONTEXT

# Why This Matters Now

This document bridges high-level experience principles with technical implementation standards and ethical governance. It's intentionally hybrid: some sections describe proven patterns, others propose emerging standards, all grounded in real deployment challenges.

THE FRAMEWORK

**Part I** — **Core Principles**
5 design patterns for agentic interfaces

**Part II** — **Taxonomy of Agency**
4-class responsibility ladder

**Part III** — **Technical & Ethical**
Standards for governance & trust

**Part IV** — **Social Calibration**
Persona & communication tuning

# 1. Goal-First Onboarding

## Outcome over Feature

Traditional UI teaches users where buttons are. Agentic experience design teaches users what outcomes are possible.

### THE PATTERN: GOAL TEMPLATES

Replace blank search bars with intent-driven starters that simultaneously define capability and constraint.

### USE CASE: FINANCIAL PLANNING AGENT

Instead of:

*"What would you like help with?"*

**Offer:**

- "Ensure my liquidity for Q1"
- "Find tax-loss harvesting opportunities"
- "Alert me to unusual spending patterns"

Each template is a contract — it tells the user what the agent can do and establishes boundaries.

# 2. Shared Autonomy

## The Variable Leash

Trust is a dynamic variable. Users must adjust machine agency based on perceived mission risk.

**WATCH** Low Risk

Agent acts; user observes real-time activity log

**ASSIST** Medium Risk

Agent proposes; user approves each high-risk node

**AUTONOMOUS** High Trust

Agent executes; provides post-action audit

**USE CASE: CODE REVIEW AGENT**

The same agent, three trust postures. User adjusts based on stakes (fixing typos vs. refactoring authentication logic).

# 3. Parallel Agency

## The Dual-Track UI

Humans and agents now operate in the same workspace simultaneously. Design must prevent "UI Collision."

**Activity Heartbeats:**
Ambient indicators (pulsing borders, subtle glows) on objects being modified

**Conflict Resolution:**
If human edits a field the agent is "ghost-editing," agent immediately yields, reverts unsaved change, moves to Wait state

**USE CASE: DOCUMENT COLLABORATION**

1. User writes introduction to report
2. Agent simultaneously generates data tables in Section 3
3. Pulsing outline indicator shows agent working on Section 3
4. User edits Section 3 → agent pauses, saves draft to sidebar, waits

# 4. Outcome Projection

## The Simulation Standard

In distributed systems, "Undo" is often a myth. "Simulation" is the only true safety net.

### THE PATTERN: GHOST STATES

Before committing actions, project the future state. Distinct color palettes signal impact.

| | |
|---|---|
| **Emerald** | Gains / additions |
| **Amber** | Losses / flagged items |
| **Line-through** | Replaced values |

### USE CASE: EMAIL CLEANUP AGENT

*"Archive all newsletters from last year"*

| | |
|---|---|
| **847 emails** | → Archive |
| **12 receipts** | → Flagged for review |

Current inbox: 1,240

**Projected: 393**

User approves or refines before execution

# 5. Progressive Auditability

## Forensics

Users must peel back layers of agentic logic as needed — from summary to raw machine data.

**LEVEL 1**

### Summary

Natural language explanation
"I moved $500 to checking
to avoid a late fee"

**LEVEL 2**

### Trace

Chronological sub-steps,
model reasonings,
API calls

**LEVEL 3**

### Raw Log

Unfiltered JSON/model
output for expert
forensic review

**USE CASE: EXPENSE APPROVAL AGENT**

CFO sees: "Approved 23 expenses, flagged 4 for review" — then drills into any flagged item across all three levels.

# The Responsibility Ladder

Interfaces must explicitly declare and visualize the "Class" of power granted to an agent.

| CLASS | NAME | SCOPE | AUTHORIZATION | USE CASE |
|---|---|---|---|---|
| **Class 1** | **Micro-Tasking** | Single, non-destructive tasks | Implicit / session | *"Format this table"* |
| **Class 2** | **Human-in-Loop** | Multi-step flows with checkpoints | Explicit per-node | *"Schedule meetings across 5 timezones"* |
| **Class 3** | **Independent** | Autonomous execution in sandbox | Goal-based Red-Line Contract | *"Audit codebase for security issues"* |
| **Class 4** | **Legal Persona** | Binding authority / proxy | MFA + Cryptographic Attestation | *"Accept terms of vendor contract"* |

### EXAMPLE: TRAVEL BOOKING AGENT EVOLUTION

- **Class 1:** "Find flights under $400"
- **Class 2:** "Find flights, propose itinerary, I'll approve purchase"
- **Class 3:** "Book travel within policy, notify me when done"
- **Class 4:** "Book travel, sign hotel contract on my behalf"

# Standards for Trust

### 6. Model Persona & Governance Disclosure

Users have a right to know "who" is doing the work. Every step shows Model ID and discloses optimization bias ("Optimized for Cost" vs. "Optimized for Accuracy").

### 7. The Authorization Contract

"Consent" is a dynamic contract, not a checkbox.

The Red-Line Contract explicitly lists: what is AUTHORIZED, what is PROHIBITED, and temporal consent that expires after goal completion.

### 8. Identity & Proof of Proxy

For Class 4 actions, provide a "chain of custody" for intent. Cryptographic Attestation badge linking to signature verified by user's secure enclave.

### 9. Memory Sovereignty

Users must own the "mental model" an agent builds of them. Memory Scrub lets users view and delete learned patterns. Context Injection pins hard truths that override model training.

### 10. Resilience & Interruptibility

Persistent, global "Kill-Switch" to pause or abort all active agents instantly. Degraded Mode UI: if service fails mid-mission, display "Last Known Good State" and manual recovery path.

### 11. Inter-Agent Coordination

When Agent A hands off to Agent B, visualize context transfer so user understands the "bridge." User can inspect/edit the handoff payload before the second agent proceeds.

### 12. Resource Transparency

Display "Fuel Gauge" showing estimated token/API costs before initiating massive Class 3 background tasks.

# The Social Scale

Allow users to set agent persona from Instrumental (concise, code-like) to Relational (conversational, high empathy). Anthropomorphism is an opt-in toggle.

**INSTRUMENTAL**

Developer context

```
> 3 vulnerabilities found.
> Fix?  [Y/n]
```

**RELATIONAL**

Non-technical context

I found three security issues in your code. The most important one could let attackers access user data. Would you like me to explain each one?

# Open Questions

Some patterns are proven in production (Parallel Agency, Outcome Projection). Others are proposed standards waiting for field validation. These are the questions we're actively testing:

### Control & Scope

- How granular should the Leash be?
- Can the Red-Line Contract scale to complex multi-agent workflows?

### Memory & Privacy

- What's the right default for Memory Sovereignty?
- How do we handle cross-context contamination?
- What's the retention policy when an agent "forgets"?

### Identity & Representation

- When an agent acts "as you," what identity does it carry?
- How do recipients know it's an agent?
- How do you audit agent actions across platforms?

### Multi-Agent Accountability

- When Agent A delegates to B to C, who's responsible?
- What's the right model for "agent reputation"?

### Cross-System Action

- What protocol governs agent-to-agent exchange?
- How do we prevent agent impersonation attacks?

TOUCH GRASS CONSULTING × TACIT.GLOBAL

# Let's build the interface layer that earns trust.

If you're building agentic systems and wrestling with these problems, I'd value your perspective. This framework is most useful when it's battle-tested against real implementation constraints.


**JAMES HALIBURTON**

Email james@touchgrass.consulting

Web www.touchgrass.consulting

Partner www.tacit.global


Findings from building production agentic systems — navigating the gap between AI capabilities and deployment reality.


DRAFT V0.2 | February 2026 | Active Working Document