Design value preservation protocol

Keeping the thinking in the design process

Version: 1.0
Date: 2026-02-27
Companion to: UX Practice OL Diagnosis v2.0, Whitepaper v2.0
Purpose: Operational checklist ensuring OL-governed design values survive the pressures of sprint cycles, stakeholder expectations, and delivery velocity.
Use: Apply at five project milestones. Any "No" answer requires a documented decision — not necessarily a fix, but a conscious choice.

The problem this solves

The UX Practice Diagnosis (v2.0) documents how the system turns methods into rituals — how sprint pressure, process compliance, and production overhead structurally prevent designers from doing the strategic work the discipline exists for. When AI compresses the production layer and frees 25-30 hours per week, the same systemic forces will try to fill that capacity with more production, faster timelines, or reduced headcount.

This protocol is designed to prevent that compression. It operationalizes the seven core design values from the OL Framework into questions that can be asked at five project milestones. Its purpose is not to add process — designers already have too much of that. Its purpose is to protect the *thinking* at the moments where the system is most likely to squeeze it out.

The protocol works through one mechanism: making value trade-offs visible. Every "No" answer is a place where the design has compromised a core value. Sometimes that compromise is necessary and justified. But it must be *chosen*, not *defaulted into* because nobody asked the question.

The seven values (Summary)

For full context, see UX Practice OL Diagnosis v2.0, Part 10.

#	Value	One-Line Test
V1	**Sovereignty as Primary Outcome**	Will the user be more capable after this journey?
V2	**Friction is a Design Material**	Have we distinguished productive friction from overhead?
V3	**The Journey is the Unit of Design**	Are we designing the journey or just the screen?
V4	**Temporal Integrity**	Does this work at day 90, not just day 1?
V5	**Verification Structurally Enabled**	Can the user evaluate whether the output is correct?
V6	**Cross-Boundary Continuity**	What happens at the transitions we don't own?
V7	**Honest Measurement**	Are we measuring capability growth or just satisfaction?

The five milestones

Milestone 1: Brief review

When the project starts. Before any design work begins.

This is the most important milestone — the one most often skipped. The UX Practice Diagnosis identifies the "pre-design failure pattern": by the time a designer receives a brief, strategic decisions have already been made. This checkpoint asks whether the brief itself preserves the conditions for good design.

Value	Question	If No
V1	Does the brief define a desired *user capability outcome*, not just a feature to deliver?	The brief is scoped to a feature. Push back: "What should the user be able to do independently after using this?" If this question can't be answered, the feature may not need to exist.
V2	Does the brief distinguish between friction the user should experience (learning, verification, judgment) and friction to remove (overhead, coordination, waiting)?	All friction is treated as bad. Flag this early: some of what looks like "pain points" may be productive difficulty.
V3	Does the brief scope the *journey* or just the *screen*? Does it account for what happens before and after the user reaches this feature?	The brief scopes a single view or flow in isolation. Ask: "What is the user doing 5 minutes before they reach this? What do they do next? What context do they carry in?"
V4	Does the brief include temporal requirements — not just "does it work on first use" but "does it still work at sustained use"?	No temporal dimension. Add: "How will we know this still serves users after 3 months of regular use?"
V5	If AI is involved, does the brief require that users can evaluate the AI's output?	AI output is presented as authoritative. Flag: "Where in this flow does the user verify rather than accept?"

Value	Question	If No
V6	Does the brief acknowledge the boundaries this feature creates or crosses — transitions to/from other tools, other features, other contexts?	Feature is designed as if it exists in isolation. Map the boundaries: "What does the user lose when they arrive here? What do they lose when they leave?"
V7	Does the brief define success metrics that include capability growth, not just task completion or satisfaction?	Success is defined as adoption rate, NPS, or time-on-task only. Propose additional metric: "What would tell us the user is getting *better* at this, not just faster?"

The Brief Review Principle: If the brief doesn't protect these values, the design won't either. The cheapest place to preserve design quality is before the first sketch. Every compromise at the brief stage compounds through the entire project.

Milestone 2: Concept review

When the design direction is established but before detailed production begins.

This is where the three-layer process collision hits hardest. The concept must survive translation into development tickets while preserving its cognitive architecture. If the concept can't be explained in terms of what it does to the user's thinking — not just what it looks like — it's a screen, not a design.

Value	Question	If No
V1	Can you state, in one sentence, what capability this concept builds in the user? Not what it *does* — what it *develops*.	The concept solves a task but doesn't develop capability. Ask: "If we removed this feature after 6 months, would the user have gained anything that persists?" (The Persistence Test)
V2	Have you identified at least one point of *productive friction* in the concept — a place where the user must think, choose, or evaluate rather than just proceed?	The flow is entirely frictionless. This is a design risk, not a design win. Identify one decision point where the user's judgment is exercised and protect it.
V3	Does the concept account for the user's state *before* they arrive and *after* they leave? Have you mapped at least one upstream and one downstream boundary?	The concept starts and ends at the feature boundary. Sketch the transitions: what context arrives with the user, what context they need to carry out.
V4	Have you considered how this concept performs after repeated use? Does the 50th use still serve the user, or does it become overhead?	Only first-use experience has been considered. Sketch the "experienced user" version: what changes, what fades, what should adapt.
V5	Is there a structural mechanism for verification — not just the *ability* to check, but an *affordance* that makes checking natural?	Verification is possible but requires effort. Move verification from "possible" to "prompted" — the design should make evaluation a natural step, not an effortful detour.

Value	Question	If No
V6	If this concept requires data or context from another tool/feature/system, have you designed the *boundary*, not just the *destination*?	Only the destination has been designed. The boundary is where context dies. Design the bridge: what transfers, what gets reconstructed, what the user must supply.
V7	Have you defined at least one metric that would tell you the concept is failing *even if users report satisfaction*?	Only satisfaction and usage metrics are planned. Add a capability indicator — this is what distinguishes OL-governed measurement from traditional UX.

The Concept Review Principle: The concept is the last moment where the cognitive architecture is visible as a whole. Once it fragments into tickets, the journey-level logic becomes invisible. Protect it here.

Milestone 3: Design review

When the detailed design is complete and ready for refinement/handoff.

This is where traditional design reviews live — and where they typically operate at the surface level (alignment, color, copy consistency). This checkpoint adds the cognitive layer that Figma can't show.

Value	Question	If No
V1	Does the detailed design still deliver the capability outcome defined at concept stage, or has it been narrowed to feature delivery?	Scope has narrowed to "does it work" rather than "does it develop." This is the most common drift. Re-read the concept review's V1 answer and check whether it still holds.
V2	Is productive friction still present in the design, or has it been smoothed out during detail work?	Friction removed during production for consistency or elegance. Check: was the friction removed because it was overhead, or because smooth flows feel more "polished"? Polish that removes productive friction is a net loss.
V3	Does the design handle the journey transitions identified at concept stage? Are the boundary experiences designed, not just the destination?	Boundaries were deferred to "future work." At minimum, document what context the user needs at entry and what they carry at exit. If you can't design the boundary, at least make it visible.
V4	Have you sketched the "day 90" version of this design — how it behaves for a user who's used it 50 times?	Only "day 1" experience exists. Temporal design doesn't require building two versions. It requires *thinking* about two states. Where does the scaffolding fade? Where does the shortcut emerge?
V5	For every AI output in the design, can you point to the specific UI element that supports user evaluation of that output?	AI output is displayed without evaluation support. Add at minimum: the source of the output, the confidence or basis, and the path to override. Verification is structural, not optional.

Value	Question	If No
V6	Has the Cx cost of this design been estimated — what does this add to the user's cross-boundary load when they transition to/from this feature?	Boundaries not assessed. Use the simple test: what does the user have to *remember* or *reconstruct* when crossing in or out? That's the Cx cost.
V7	Beyond task completion metrics, what specific user behavior would indicate this design is *succeeding* at building capability? What behavior would indicate it's *failing*?	No behavioral indicators defined. Define at least one positive signal (user chooses to verify, explores alternatives, adapts the tool to new situations) and one negative signal (user accepts all defaults, never overrides, uses the feature less skillfully over time).

The Design Review Principle: The traditional design review checks whether the design looks right. The OL review checks whether it *works right* — at the cognitive level, over time, across boundaries.

Milestone 4: Pre-handoff

After the design is final, before it enters development.

This is the last moment where the cognitive architecture is in the designer's hands. Once it's in tickets, the journey fragments. This checkpoint protects the values that are most vulnerable to implementation translation.

Value	Question	If No
V1	Have you communicated the *sovereignty intent* to the development team — not just what to build, but what capability it should preserve in the user?	Only specification was handed off. Add a one-paragraph "Design Intent" to the handoff: "This feature is designed to develop [capability]. The critical path is [verification step / friction point / sovereignty checkpoint]. Please preserve this in implementation."
V2	Are the points of productive friction explicitly marked in the specification so development doesn't optimize them away?	Friction points are indistinguishable from implementation constraints. Annotate them: "This step is intentionally effortful. Do not auto-complete, pre-fill, or skip." Without this annotation, a conscientious developer will "fix" the friction.
V3	Does the handoff include the journey context — what happens before/after — or only the feature specification?	Only the feature is specified. Include a journey snippet: one screen before, one screen after, with boundary notes. This is the minimum context a developer needs to implement transitions correctly.
V4	Are temporal behaviors specified — not just initial state, but how the feature should adapt with repeated use?	Only initial state specified. If the feature should behave differently for experienced users (progressive disclosure, scaffolding fade, shortcut emergence), specify it now. If not specified, it

Value	Question	If No
		won't be built.
V5	Are verification mechanisms specified as requirements, not suggestions?	Verification elements are "nice to have." Elevate to requirements: "Citations must be inline." "Confidence indicators must be visible." "Override path must be accessible within one interaction." If verification is optional, it will be cut.
V6	Have boundary behaviors been specified — what context must be passed, what state must be preserved, what the user should see at entry/exit?	Boundaries are undefined. At minimum: specify what data travels with the user and what the entry state should be. Unspecified boundaries will be implemented as clean breaks — which is where context dies.
V7	Are the capability metrics included in the acceptance criteria, or only functional requirements?	Only functional acceptance criteria exist. Add at least one behavioral acceptance criterion: "User can [perform verification / exercise judgment / modify output] within the flow." This makes capability preservation testable.

The Pre-Handoff Principle: Everything not specified will be decided by someone who hasn't thought about it. Protect the values by making them explicit in the specification — especially the ones that look like "inefficiencies" to a developer optimizing for clean code.

Milestone 5: Post-launch review

After the feature has been live long enough for temporal patterns to emerge (minimum 4-6 weeks).

This is the milestone that almost never happens in sprint-driven practice. It's also the only one that can detect Ct (temporal degradation) and confirm whether V1 (sovereignty) is actually being achieved.

Value	Question	If No
V1	Are users demonstrating increased capability — better judgment, more independent thinking, stronger evaluation — compared to pre-launch?	No capability growth detected, or not measured. This is the governance question. If the answer is "we don't know," the measurement system failed. If the answer is "no," the design needs revision.
V2	Is the productive friction still functioning, or have users found workarounds that bypass it?	Users bypass friction points. Evaluate: are they bypassing because they've developed the skill (scaffolding fade — good) or because they've found shortcuts that avoid the thinking (friction avoidance — bad)?
V3	Are the journey transitions working — do users maintain context across boundaries, or do they arrive disoriented?	Context loss at boundaries. This is Cx data. Identify the worst boundary and design a bridge for the next iteration.

Value	Question	If No
V4	Has output quality degraded over time — for users, for AI outputs, for the experience as a whole?	Temporal degradation detected. This is Ct in action. Identify whether the degradation is in the system (AI output drift), the user (calibration shift, verification decline), or both. Each requires a different intervention.
V5	Are users still verifying AI outputs, or has verification declined over time?	Verification declining. This is the automation complacency pattern. The verification affordance may need strengthening, or the design may need periodic re-engagement mechanisms.
V6	Have new boundaries emerged that weren't anticipated — integrations, workarounds, or adjacent tools that create unexpected Cx costs?	Unanticipated boundaries exist. Map them. They're part of the real journey now, whether designed or not.
V7	Do the capability metrics show a trajectory, or just a snapshot? Can you see whether users are *developing* over time, not just performing?	Only snapshot data available. Longitudinal measurement requires baseline comparison. If you can't see the trajectory, you can't see temporal degradation or capability growth. This is the measurement gap to close first.

The Post-Launch Principle: The design isn't done when it ships. It's done when you know whether it made users better. If you never check, you'll never know — and the temporal effects (complacency, calibration drift, skill atrophy) will accumulate silently.

How to use this protocol

Minimum viable application: Pick one milestone. Use it once. See what it surfaces. The Brief Review (Milestone 1) is the highest-leverage single checkpoint because it catches problems before they propagate.

Full application: Use all five milestones across a project. Document answers. Review the pattern of "No" answers across milestones — they'll reveal which values your organization structurally resists preserving. That pattern is the OL diagnosis of your team's process.

Team application: Run the milestone questions as a 15-minute check at existing ceremonies. The Brief Review fits in kickoff. The Concept Review fits in design critique. The Design Review fits in the existing review. The Pre-Handoff fits in refinement. The Post-Launch fits in retrospective. No new meetings required — just new questions in existing ones.

The meta-test: If your team doesn't have time for a 15-minute check at any of these milestones, that itself is the diagnosis. The system has consumed so much capacity that even protecting the thinking requires more capacity than is available. That's the condition the OL Practice Diagnosis describes — and it's the condition AI compression is about to

change.

One page summary

Print this. Tape it to the wall.

Milestone	When	Core Question
Brief Review	Project start	Does this brief protect the conditions for good design?
Concept Review	Direction set	Does this concept build capability, not just deliver a feature?
Design Review	Detail complete	Does this design preserve its cognitive architecture?
Pre-Handoff	Entering development	Are the values explicit in the specification?
Post-Launch	4-6 weeks live	Did this design make users better?

At every milestone, ask: Is the user more capable because of this design — or just more efficient?

If you can't answer that question, the design isn't done.

Document Version: 1.0.0
Created: 2026-02-27
Framework Reference: OL Framework Parts 1-5 (v1.1-v1.5), Whitepaper v2.0
Companion: UX Practice OL Diagnosis v2.0