Decision-OS v4 Addendum: Grounded Decision Protocol (V5-compatible Guard)

SiriusA Adoption Gate (0.3 Disclosure)

Hold-First Auditing for Irreversible Actions

+ V4-Compatible Fit Check (Appendix)

Shinichi Nagata

February 3, 2026

Abstract

This *Decision-OS v4 Addendum* publishes a grounded, citable decision protocol as a minimal *0.3-disclosure* interface: inputs, computation skeleton, decision labels (GO/HOLD/NO), and a reproducibility pack (schema, demo tests, and reference pseudocode). The protocol is *V5-compatible* in the sense that it supports guard-first auditing for irreversible actions, while keeping calibration-sensitive operating details undisclosed (0.7), including real thresholds, update rules, exception criteria, and the data-collection protocol.

1 Disclosure Boundary (0.3 / 0.7)

Disclosed (0.3). Interface format, computation skeleton (non-calibrated), decision labels (GO/HOLD/NO), and a minimal reproducibility pack (schema + demo test vectors + reference pseudocode).
Not disclosed (0.7). Real operating thresholds, calibration/update rules, exception criteria, and data-collection protocol.

2 Positioning (One-line Map)

This addendum publishes a grounded decision protocol that is readable and citable as a *spec*, while keeping calibration-sensitive operating parameters undisclosed.

3 Decision Labels (UI)

We use **GO / HOLD / NO** as the reader-facing labels:

- **GO**: execute now (conditions satisfied).
- **HOLD**: do not execute yet (needs verification / more data / time).
- **NO**: do not execute (reject / exit).

4 Protocol Spec (V4 Core + V5-compatible Guard)

4.1 Inputs

Let the inputs be: h_s (hours saved), h_m (maintenance hours), w (unit cost per hour), CVR, g (gross profit per conversion), and n (volume). Set $k = 4$ as the scale factor. If any required input is missing, the default output is **HOLD**.
If uncertainty dominates (inputs are not verifiable), default to **HOLD** and state the missing evidence in one line.
CVR note. Use a probability in $[0, 1]$ (e.g., 0.06 for 6%), not a percent integer.

4.2 Practical Benefit (3-scenario)

For each scenario $\omega \in \{\text{downside}, \text{baseline}, \text{upside}\}$:

$$\text{Practical}(\omega) = (h_s(\omega) \cdot w \cdot k) + (\text{CVR}(\omega) \cdot g(\omega) \cdot n(\omega)) - (h_m(\omega) \cdot w \cdot k).$$

4.3 EV and Weighted Expectation

Define:

$$\text{EV}(\omega) = \text{Practical}(\omega), \qquad \mathbb{E}[\text{EV}] = 0.2 \cdot \text{EV}(\text{downside}) + 0.6 \cdot \text{EV}(\text{baseline}) + 0.2 \cdot \text{EV}(\text{upside}).$$

4.4 Decision Gate (V5-compatible Guard)

IF Practical(downside) < 0 **THEN NO** (provide a one-line reason).
Let h_{mid} and h_{ev} be *user-supplied thresholds* (calibration undisclosed in this document).

> **IF** Practical(baseline) $\geq h_{\text{mid}}$ **AND** $\mathbb{E}[\text{EV}] \geq h_{\text{ev}}$ **THEN GO**
> **ELSE HOLD/NO** (provide a one-line reason).

Note. In code/JSON we use `h_mid`, `h_ev`, corresponding to h_{mid}, h_{ev} in math.

5 Premortem (Cause / Signal / Avoidance) x3

- (1) Cause: Signal: Avoidance:
- (2) Cause: Signal: Avoidance:
- (3) Cause: Signal: Avoidance:

6 How to Use (Minimal Workflow)

This protocol is intentionally *non-calibrated*. Users provide their own thresholds (`h_mid`, `h_ev`) and refine them over time.

1. **Define the decision.** Specify the action that would be taken if the output is **GO**.
2. **Fill three scenarios.** Estimate inputs for **downside / baseline / upside**.
3. **Compute.** Calculate Practical(ω) and $\mathbb{E}$[EV].

4. **Set thresholds.** Choose (`h_mid`, `h_ev`) as user-defined criteria.

5. **Gate.** Output **GO** if the gate passes; otherwise output **HOLD/NO** with a one-line reason.

6. **Premortem.** If mitigation is not concrete, keep **HOLD**.

7. **Re-review.** For **HOLD**, state the missing evidence in one line and re-run after new information arrives.

Note. Thresholds are not "magic numbers"; they represent the user's cost-of-error and irreversibility tolerance for the given domain.

A Repro Pack (Schema + Demo Tests + Reference Pseudocode)

A.1 A1. Input Schema (JSON example)

```
{
  "scenario_keys": ["downside", "baseline", "upside"],
  "hs_hours_saved": {"downside": 0, "baseline": 0, "upside": 0},
  "hm_maint_hours": {"downside": 0, "baseline": 0, "upside": 0},
  "w_cost_per_hour": 0,
  "cvr_prob_0to1": {"downside": 0.0, "baseline": 0.0, "upside": 0.0},
  "g_gross_profit_per_conversion": {"downside": 0, "baseline": 0, "upside": 0},
  "n_opportunities": {"downside": 0, "baseline": 0, "upside": 0},
  "k_scale_factor": 4,
  "thresholds": {"h_mid": null, "h_ev": null}
}
```

A.2 A2. Demo Thresholds (for examples only)

For the demo cases below, set:

$$h_{\text{mid}}^{\text{demo}} = 50{,}000, \qquad h_{\text{ev}}^{\text{demo}} = 80{,}000.$$

These values are illustrative and not operational calibration.

A.3 A3. Demo Test Vectors x3

A.3.1 Case 1: GO (clear pass)

Set $w = 5{,}000$ and $k = 4$.

$$(h_s, h_m, \text{CVR}, g, n)_{\text{downside}} = (2, 1, 0.40, 30{,}000, 10)$$

$$(h_s, h_m, \text{CVR}, g, n)_{\text{baseline}} = (3, 1, 0.50, 30{,}000, 10)$$

$$(h_s, h_m, \text{CVR}, g, n)_{\text{upside}} = (4, 1, 0.60, 30{,}000, 10)$$

Then:

$$\mathbb{E}[\text{EV}] = 190{,}000 \Rightarrow \textbf{GO}.$$

A.3.2 Case 2: HOLD (insufficient margin / needs verification)

Set $w = 5{,}000$ and $k = 4$.

$$(h_s, h_m, \mathrm{CVR}, g, n)_{\text{baseline}} = (2, 2, 0.15, 20{,}000, 5)$$

Then:

$$\mathbb{E}[\mathrm{EV}] = 15{,}000 \Rightarrow \textbf{HOLD}.$$

A.3.3 Case 3: NO (negative downside)

Set $w = 5{,}000$ and $k = 4$.

$$(h_s, h_m, \mathrm{CVR}, g, n)_{\text{downside}} = (1, 4, 0.05, 20{,}000, 5)$$

Then:

$$\mathrm{Practical}(\text{downside}) = -55{,}000 \Rightarrow \textbf{NO}.$$

A.4 A4. Reference Pseudocode (non-calibrated)

```
def practical(hs, hm, w, k, cvr, g, n):
    return hs*w*k + cvr*g*n - hm*w*k

def expected_ev(P_down, P_base, P_up):
    return 0.2*P_down + 0.6*P_base + 0.2*P_up

def gate(P_down, P_base, EVexp, h_mid, h_ev):
    if P_down < 0:
        return "NO", "Downside scenario negative."
    if (P_base >= h_mid) and (EVexp >= h_ev):
        return "GO", "Baseline & expected EV exceed thresholds."
    return "HOLD", "Threshold not met; verify inputs or redesign."
```