UXL FOUNDATION

UXL Skills Evaluator

Project-maintained operational knowledge for coding agents

Maintainer overview · September 2026

What exists today

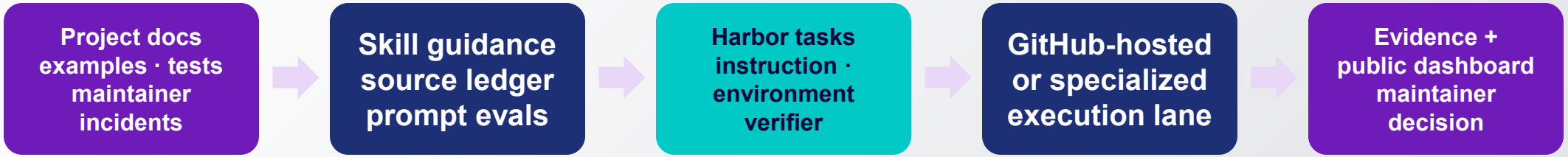
- **8 skills** — Six library portfolios plus shared SYCL build/debug and performance-validation guidance.
- **52 scoped evaluation tasks** — 41 implemented and 11 left visible as planned gaps; this is inventory, not a pass-rate claim.
- **A public control room** — Skills, evaluations, platforms, methodology, maturity, source freshness, and ownership status.
- **A portable execution model** — GitHub-hosted checks by default; opt-in specialized machines only when a task contract requires them.

The current skill catalog — at a glance

- **oneDNN** — Primitives, layouts/reorders, fusion, numerics, benchdnn.
- **oneMath** — Domains, dispatch modes, queues, backends, linking.
- **oneDAL** — Python/native paths, modes, tables, metric parity, conversion cost.
- **oneTBB** — Patterns, shared state, flow graphs, arenas, grainsize/affinity.
- **oneDPL** — Host/device policies, iterators, lifetime, synchronization, ordering.
- **oneCCL** — Collectives, ranks, launch symmetry, waits, plugins, hangs.
- **Cross-project** — SYCL build/runtime triage and correctness-first performance evidence.

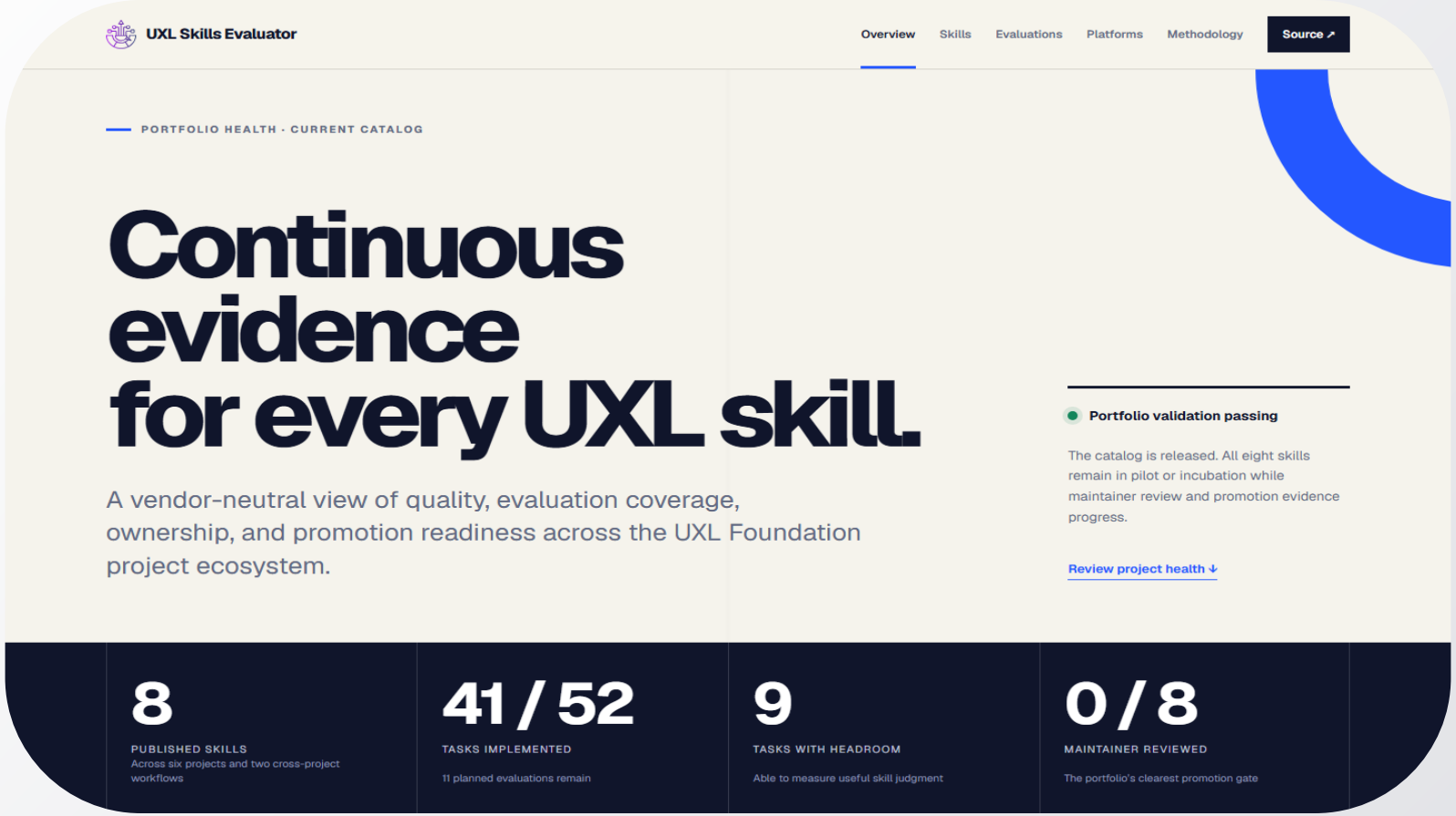
Architecture: project knowledge becomes measured evidence

One repository carries guidance, task contracts, policy, and the public portfolio view.

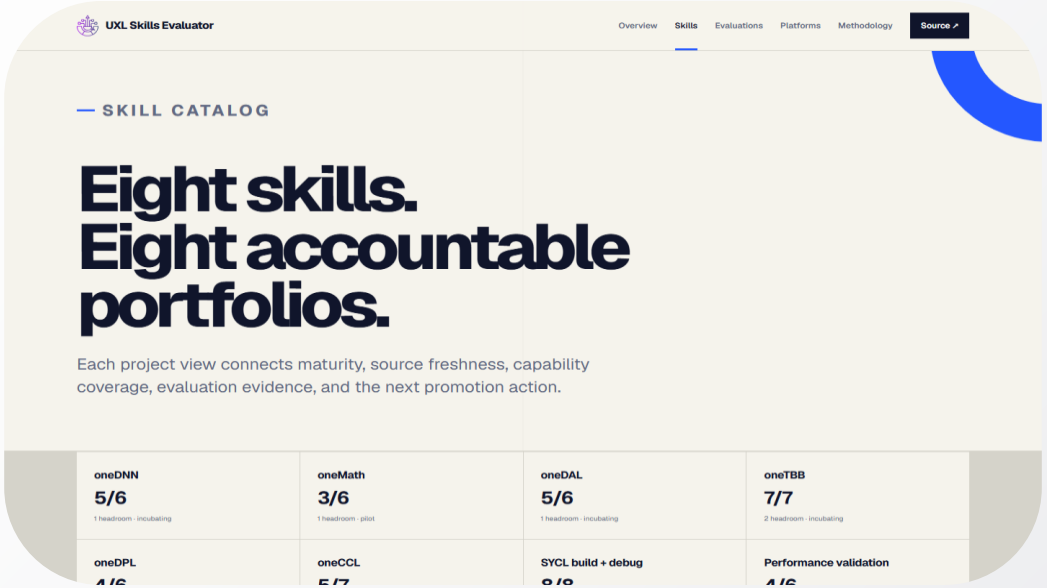


Specialized machines change the environment, not the evidence contract.

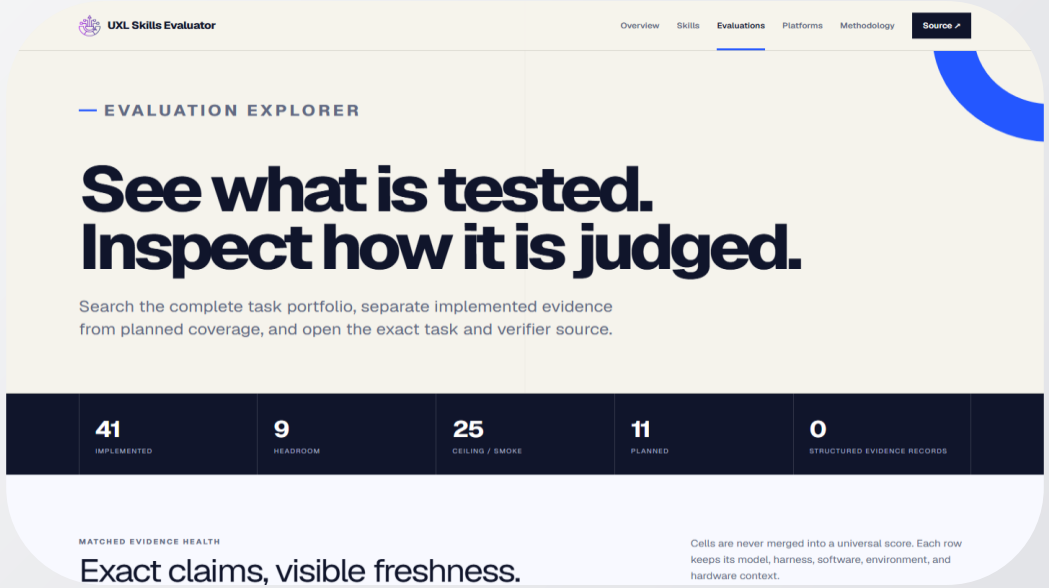
The public control room answers: are the skills healthy?



Maintainers can drill from portfolio to exact task contracts

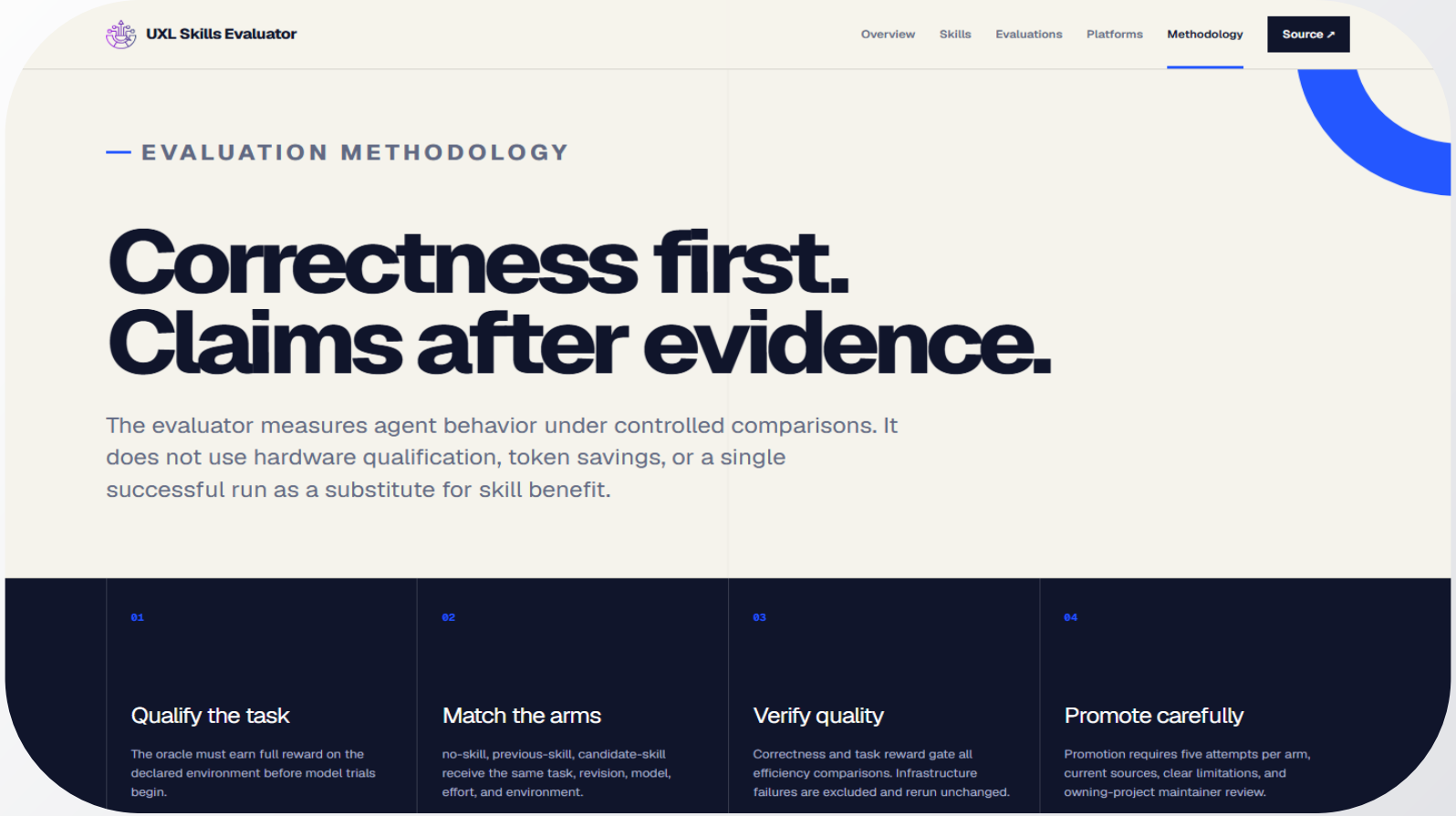


Skills: scope, maturity, sources, limitations



Evaluations: exact tasks, states, environments

The methodology is deliberately claim-resistant



Every claim lives inside a matched evaluation cell

Inside one cell, change only the skill treatment.

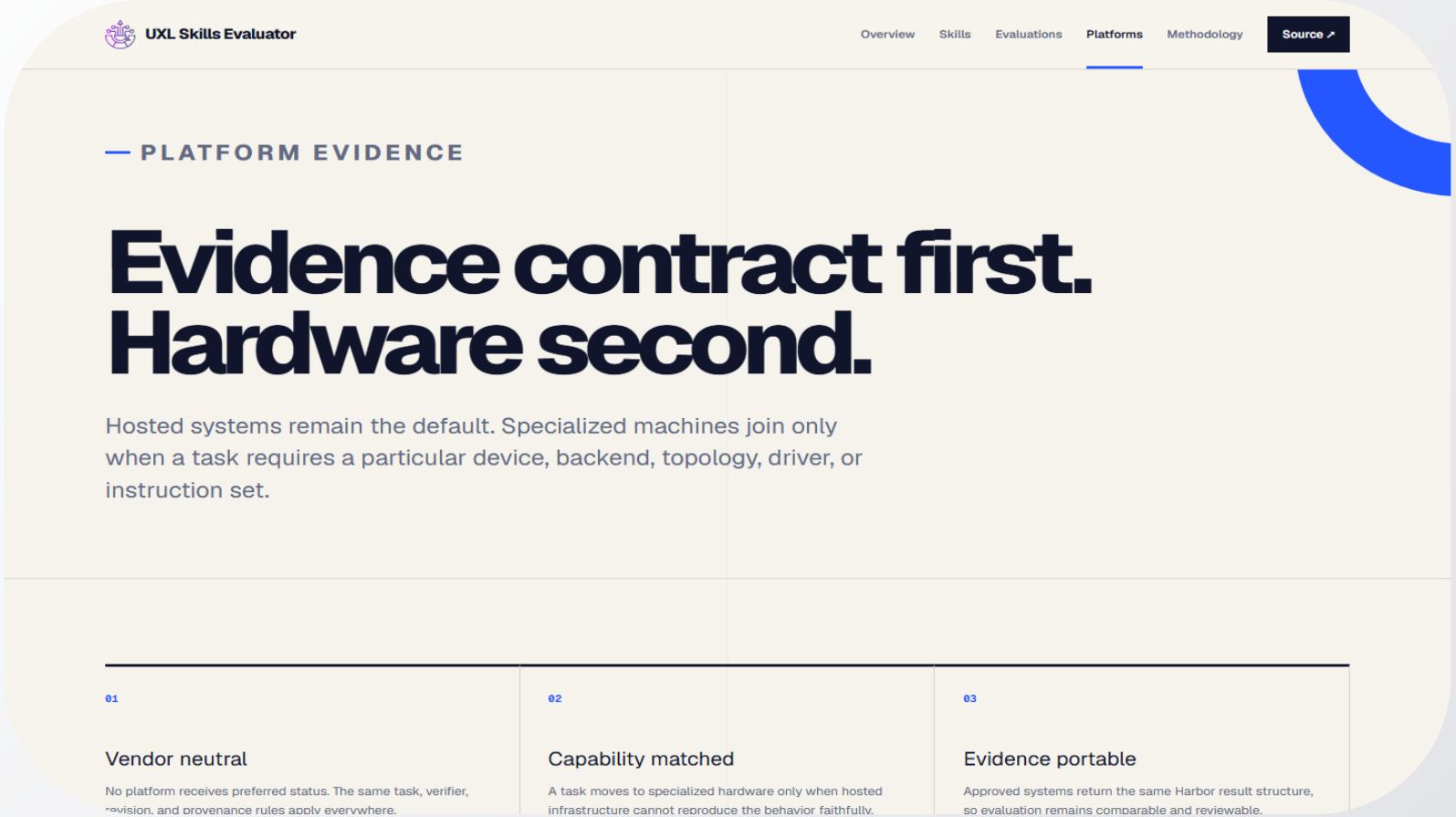


Repeat across representative cells; report differences. Never average incompatible contexts into a universal score.

Maintainer pushback is expected — and the answer is scoped evidence

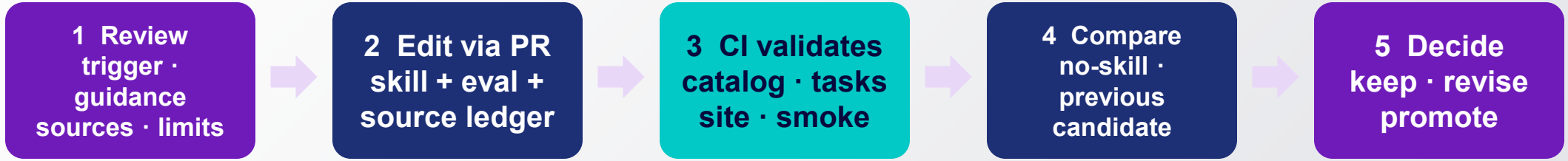
- **Do you test every combination? — No. We publish representative, risk-based cells and the gaps.**
- **What changes invalidate evidence? — Re-run affected cells; older results remain history, not current proof.**
- **How do you limit overfitting? — Maintainer incidents, hidden checks, negative controls, multiple tasks, and trajectory review.**
- **What if every arm passes? — Keep it as smoke coverage; claim no lift and seek a harder task.**
- **Does hardware qualification prove skill value? — No. Qualification and matched skill trials are separate gates.**
- **Can scores be pooled? — No universal score; report conclusions within each model, harness, software, and environment cell.**

Hardware is a pluggable evidence lane — never the product



The ongoing maintainer workflow fits normal GitHub practice

A maintainer's recurring action is a small review — the shared evaluator does the rest.



Quality is the gate; lower token use never compensates for fewer verified successes.

Ownership is intentionally split

- **Project maintainers own** — Triggers, technical guidance, official sources, limitations, and whether task scenarios are authentic.
- **UXL shared infrastructure owns** — Catalog validation, Harbor harnesses, comparison policy, dashboards, hosted CI, and runner contracts.
- **Hardware owners opt in** — They qualify a declared environment, run only reviewed immutable evaluator revisions, and return evidence artifacts.
- **No silent promotion** — Project approval, source verification, realistic eval prompts, matched trials when feasible, and explicit limitations remain gates.

The decision we need from each project

- **Name one owner** — A maintainer or small reviewer group for scope and source freshness.
- **Approve or correct the first-pass guidance** — Especially triggers, gotchas, unsupported paths, and terminology.
- **Validate the task inventory** — Keep, rewrite, or delete scenarios; nominate one real support issue or regression.
- **Adopt a light cadence** — Review skill-affecting PRs and recheck sources at releases or quarterly.
- **Start small** — A 30-minute review is enough to turn this from UXL-authored draft into project-backed guidance.