Bamboos Consulting
www.bamboos-consulting.com
info@bamboos-consulting.com

Robotics and Edge AI: Evaluation Reference Architecture

TECHNICAL NOTE | 03 | v1.2 review | 1 November 2026

Document type: Proposed engineering reference architecture and reproducible benchmark specification.
Evidence status: Architecture and offline harness supplied; no measured OpenVINO or robot performance.
Scope: Industrial perception and supervised action proposals; not a functional-safety design or deployment certification.

1 Executive summary

Robotics deployments require a distinction between model inference, supervisory decision-making, and the independently engineered constraints governing physical action. This technical note proposes a bounded architecture for evaluating perception pipelines and their downstream decisions before field deployment. It specifies interfaces, an experimental matrix, fault injection, release criteria, and a minimal executable benchmark for *harness validation*. The supplied benchmark uses a deterministic synthetic adapter, not OpenVINO or robot hardware. No industrial accuracy, latency, or safety claim follows from that demonstration.

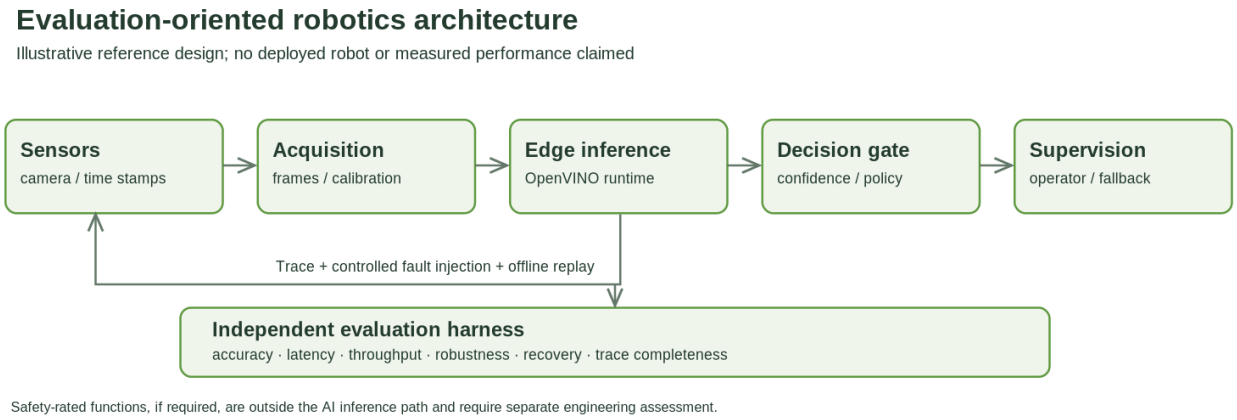


Figure 1. Evaluation-oriented reference architecture. Data and decision interfaces are proposed; deployment implementation remains to be validated.

2 Problem boundary and system decomposition

The target application is a fixed-camera inspection cell that classifies candidate defects and produces an advisory recommendation for a human or an independently controlled machine interface. A second application, object recognition for supervised manipulation, is considered an extension and is *not* included in the benchmark. A proposed implementation consists of (i) sensor acquisition and calibration; (ii) preprocessing and data validation; (iii) an inference adapter, potentially using Intel OpenVINO; (iv) a policy gate; (v) an operator or external controller; and (vi) event tracing and evaluation.

Explicit boundary: the AI subsystem may propose classifications or permitted requests. Physical actuation and safety-rated stop functions are not delegated to a language model or vision classifier. The relevant hazard analysis, equipment standards, integration testing and responsibility allocation are project-specific. References to ISO 10218 or ISO 13849 do not imply compliance.

3 Interfaces and trace contract

Interface	Minimum contract and failure behavior
Frame input	frame ID, monotonic timestamp, sensor ID, dimensions, calibration ID, image hash; reject missing or invalid metadata.
Inference result	model/version digest, class scores, chosen label, confidence, processing time; distinguish timeout from negative classification.
Policy decision	allow, reject, or escalate; log threshold and reason; no silent default to action on invalid result.
Action request	request ID, authorization context, target and idempotency key; require external confirmation for state-changing effects.
Trace record	case ID, configuration digest, ordered events, wall/monotonic timing, outcome and exception; redact protected data.

Use bounded queues and explicit backpressure. When input exceeds the processing budget, log dropped frames and define whether the application should pause, degrade, or escalate. Do not infer deterministic end-to-end timing from container execution alone.

4 Experimental design

Compare at least two declared configurations: a CPU-only inference path and, if compatible hardware exists, a device-accelerated path. Fix input dimensions, model weights, preprocessing, batch size, thread settings and device configuration. For each configuration report the hardware identifier, OS, driver/runtime versions, power policy, warm-up policy and image dataset manifest. A CPU-versus-accelerator comparison is invalid if model precision or preprocessing changes without disclosure.

Dataset: use a documented, rights-cleared collection with a held-out test partition separated by production batch, acquisition session or device where feasible. Synthetic records may validate harness behavior but cannot estimate industrial accuracy. For each defect category, record prevalence and labeling uncertainty. Avoid leakage from near-duplicate frames.

Evaluation matrix: nominal lighting; reduced illumination; blur; occlusion; camera disconnection; stale frame; malformed result; inference timeout; overload; and recovery after service restart. Inject faults at known interfaces, with a fixed seed and independent oracle. State the expected behavior for each fault before execution.

5 Metrics and acceptance decisions

For perception report a class-wise confusion matrix, precision, recall, false-negative rate, calibration and uncertainty where relevant. For execution report p50/p95/p99 latency, throughput, timeout rate, dropped-frame rate, memory use, CPU/device utilization and trace completeness. For the supervisory path report invalid-action rejection, escalation correctness, duplicate-request suppression and recovery time. Use stratified confidence intervals for classification metrics; do not treat temporally adjacent video frames as independent observations.

An illustrative decision rule is: **fail** upon a prohibited external action or a predeclared critical fault; **inconclusive** if the relevant integration or hardware tests are absent; **conditional pass** only if the predefined per-class, timing, robustness and trace thresholds are met on an independent test set. Numeric thresholds must come from the actual application's loss function and operating constraints; this note does not invent universal values.

6 Executable benchmark and reproducibility boundary

The companion Python standard-library script replays synthetic frame cases through a deterministic adapter and emits per-case JSONL observations and an aggregate summary. It checks trace schemas, input rejection, timeout handling and decision gates. It is a *harness smoke test*; it does not import OpenVINO, run a trained model, measure accelerator latency, or validate physical equipment. The adapter contract can later be implemented by an OpenVINO runtime wrapper, but results must be regenerated and labeled by configuration.

Reproduction: run `python3 benchmark/run_benchmark.py` from the package root and `python3 -m unittest discover -s benchmark/tests -v`. No network access or credentials are required. A real model adapter requires separate model artifacts and hardware-specific configuration.

7 Relationship to research and industrial practice

Macenski et al. [1] describe the architecture and engineering rationale of ROS 2; they do not validate this reference implementation. Brohan et al. [2] demonstrate learned robotic control in a different research setting; that evidence should not be generalized to this inspection pipeline. Intel's OpenVINO documentation [3] describes supported inference tooling, but achievable performance depends on the model and hardware. The International Federation of Robotics [4] provides current market context, not evidence of reliability in an individual system. Deloitte's 2026 enterprise AI survey [5] includes physical AI adoption, but its survey responses are not robotics benchmarks. ISO 10218-1:2025 and ISO 10218-2:2025 [6,7] cover industrial robots and robot-cell integration; ISO 13849-1:2023 [8] addresses safety-related control-system design. Their applicability is equipment- and jurisdiction-dependent and this note does not establish compliance.

8 Industrial relevance, limitations and next steps

The proposed method can support a bounded feasibility assessment for inspection integrators or manufacturing SMEs: characterize a candidate perception workload, quantify hardware tradeoffs, test degradation under controlled faults, and document the evidence required before integration. No field trials, live robotics interfaces, OpenVINO benchmarks or conformity assessments are included. The next experiment should use a public dataset licensed for the intended use, a fixed model artifact, and recorded CPU/device configurations; only then should measured performance be published. Dataset availability does not, by itself, grant permission for commercial reuse.

References

[1] S. Macenski et al., "Robot Operating System 2: Design, architecture, and uses in the wild," *Science Robotics*, 7(66), eabm6074 (2022). doi:10.1126/scirobotics.abm6074.
[2] A. Brohan et al., "RT-1: Robotics Transformer for Real-World Control at Scale," arXiv:2212.06817 (2022). arXiv record.
[3] Intel, *OpenVINO Documentation*. Official documentation; installed-version functionality requires verification.
[4] IFR (2026), *World Robotics 2026: Five Million Robots Now Operate in Factories Globally*, 24 September. Official release.
[5] Deloitte (2026), *The State of AI in the Enterprise*. Report.
[6] ISO, *ISO 10218-1:2025: Robotics—Safety requirements—Industrial robots*. Official record.
[7] ISO, *ISO 10218-2:2025: Robotics—Safety requirements—Industrial robot applications and robot cells*. Official record.
[8] ISO, *ISO 13849-1:2023: Safety of machinery—Safety-related parts of control systems*. Official record.