AI in Design Verification

Where It Helps, Where It Hurts, and How to Pilot Safely

May 2026

Session objective and learning outcomes

By the end of this session, engineers should be able to:

1. Recognise where AI can support the DV flow without taking control of sign-off.
2. Separate useful productivity gains from unsupported assurance claims.
3. Identify high-risk AI use cases before they reach programme-critical decisions.
4. Define a bounded, measurable and reviewable pilot for “AI in DV”.

Training stance
AI should assist bounded, reviewable tasks. It should not become an unreviewed sign-off authority.

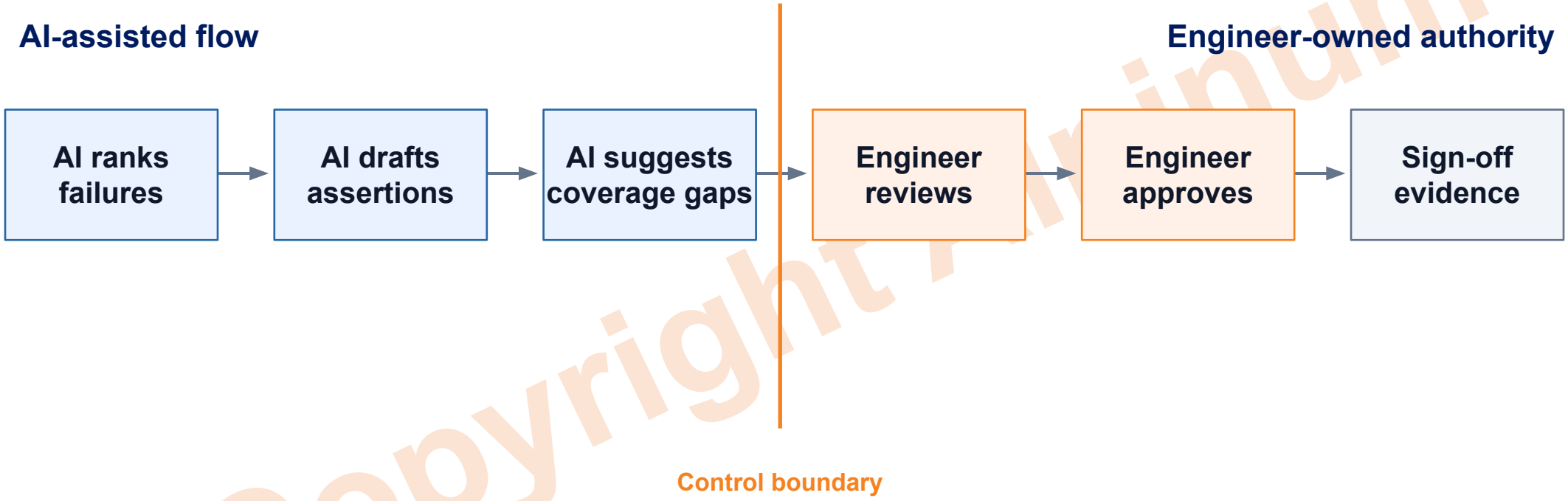
Why this matters now

The useful question is not “Can AI do verification?”

The useful question is: where can AI reduce effort, improve prioritisation and shorten debug without weakening traceability?

Complexity | **Regression cost** | **Coverage closure** | **Debug time** | **Sign-off confidence**

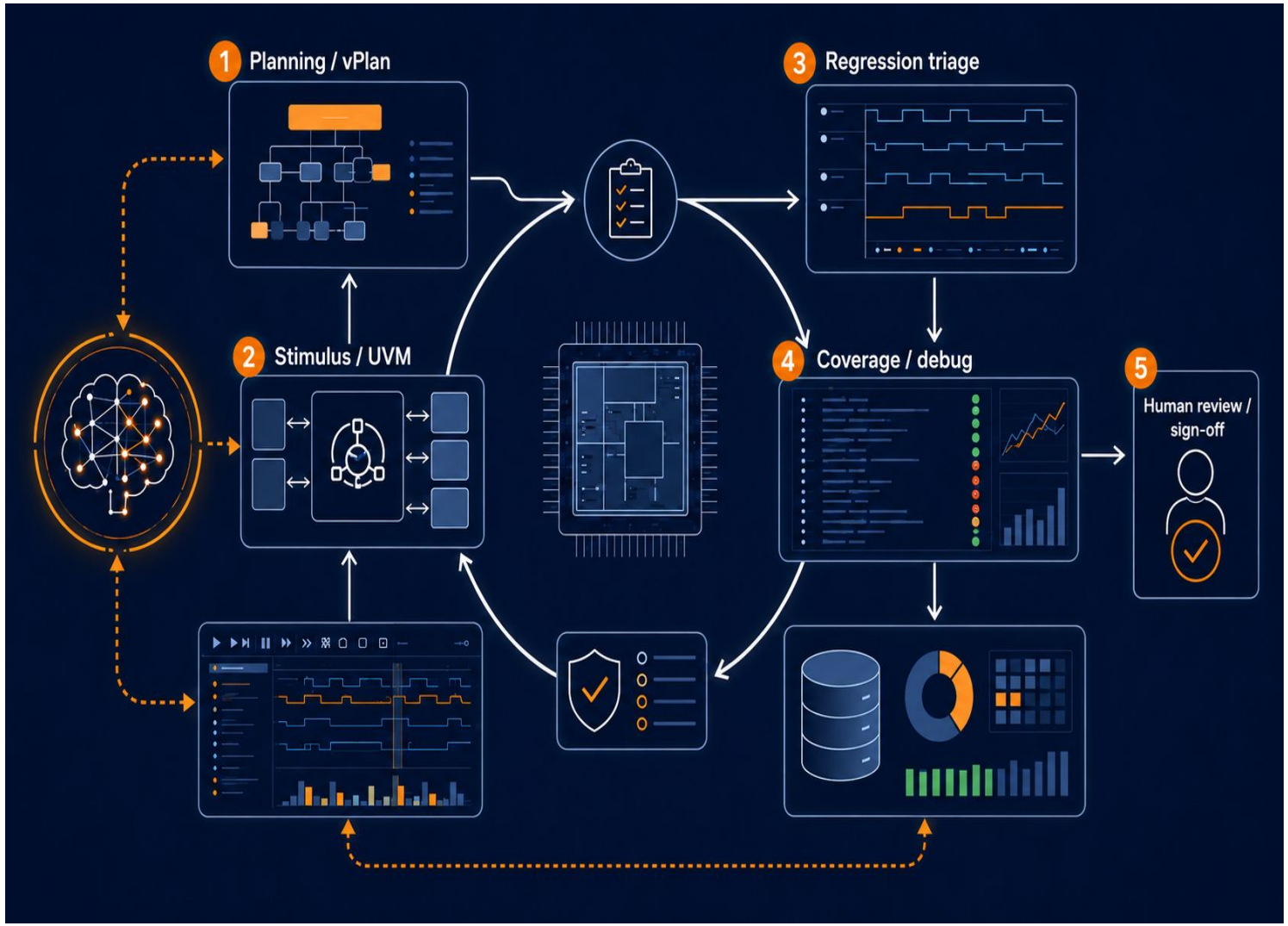
The control boundary is the central design decision



Rule for adoption

Outputs may be generated by AI, but accountable engineering decisions must remain traceable, reviewable and owned.

AI touchpoints across the verification loop



Useful insertion points

1. **Planning assistance and vPlan review**
   AI helps analyse requirements, risks and coverage intent to shape the verification plan.

2. **Stimulus generation and UVM scaffolding support**
   AI drafts testbench components, sequences and constraints to accelerate environment bring-up.

3. **Regression prioritisation and failure clustering**
   AI ranks tests, predicts failures and clusters similar failures to focus engineering effort.

4. **Coverage-gap analysis, debug support and formal assistance**
   AI identifies coverage gaps, summarises logs, suggests likely root causes and supports formal counterexample triage.

5. **Human review and sign-off control**
   Engineers review, approve and decide. Sign-off evidence and accountability remain fully human-owned.

✓ **Keep sign-off decisions outside the automation boundary.**

AI assists at defined workflow points around existing artefacts such as vPlans, testbenches, logs, coverage databases and counterexamples. **Engineering judgement and sign-off remain non-negotiable.**

Where AI helps most

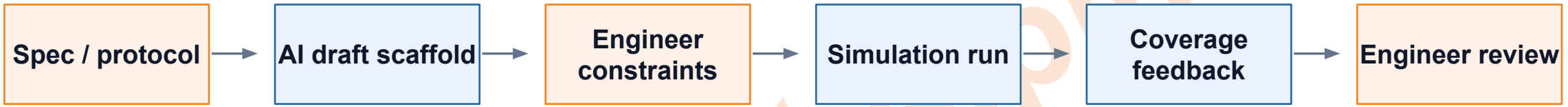
Best-fit use cases share three engineering properties:

Narrow inputs	Measurable outputs	Clear feedback
The model works from bounded artefacts, logs, protocol fragments or known environment structures.	The result can be compared with a baseline such as triage time, acceptance rate or coverage movement.	Engineers can inspect, accept, reject and improve the output through the normal review path.

Testbench assistance | **Assertion support** | **Regression triage** | **Coverage analysis** | **Debug clustering**

Use case 1: Testbench and stimulus assistance

AI can reduce setup friction, but it does not understand intent at sign-off depth.



Good fit

- UVM component scaffolding
- Template generation
- Stimulus refinement suggestions
- Coverage-directed exploration

Required controls

- Syntax and style checks
- Constraint review
- Known assumptions captured
- Baseline comparison

Do not claim

- Complete intent coverage
- Autonomous environment ownership
- Release confidence from generated code alone

Use case 2: Assertion and formal assistance

A generated assertion is only useful after semantic review.

AI can help with	Engineer must verify	Formal risk
• Candidate properties • Interface protocol checks • Template variations • Counterexample summaries	• Specification intent • Assumptions and constraints • Vacuity and completeness • Waiver logic	Small wording errors can create false confidence or over-constrained proofs

Principle: treat AI-generated properties exactly like human-authored properties until hard evidence justifies a narrower automation boundary.

Use case 3: Regression, debug and coverage analysis

This is often the safest first pilot because the data is already structured and measurable.

Regression triage	Cluster similar failures, remove duplicate investigation, prioritise likely root causes
Debug support	Summarise logs, group signatures, identify repeated failure classes
Coverage analysis	Rank gaps, highlight unhit scenarios, propose investigation areas
Test ranking	Reduce waste by prioritising tests that improve confidence or expose high-risk areas

Success signal
Less manual search effort, faster root cause learning and fewer wasted runs.

What good “AI in DV” use cases look like

Structured input

Logs, coverage databases, protocol specs, assertions, test history

Inspectable output

Rankings, drafts, clusters, summaries or recommendations

Baseline available

Human triage, existing coverage review, current regression policy

Away from sign-off boundary

The model proposes, engineers validate and approve

If any one of these is missing, pilot risk rises sharply.

Where AI can hurt a verification programme

Risk appears when attractive artefacts are mistaken for engineering evidence.

Plausible but wrong

Generated assertions or UVM code can compile while checking the wrong behaviour.

Data bias

Historical project data may teach the model the last design, not the current one.

Weak provenance

Unclear origin breaks traceability from requirement to artefact to review.

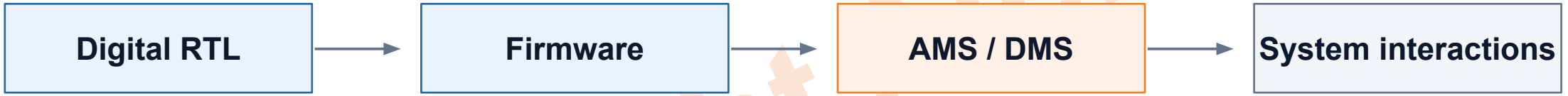
False confidence

Coverage movement or clean dashboards can look like assurance without semantic value.

Programme risk: teams gain speed but lose the evidence chain that gives verification its value.

Cross-domain verification raises the adoption risk

A model that performs acceptably in one digital block flow may not generalise to mixed-domain or system-level behaviour.



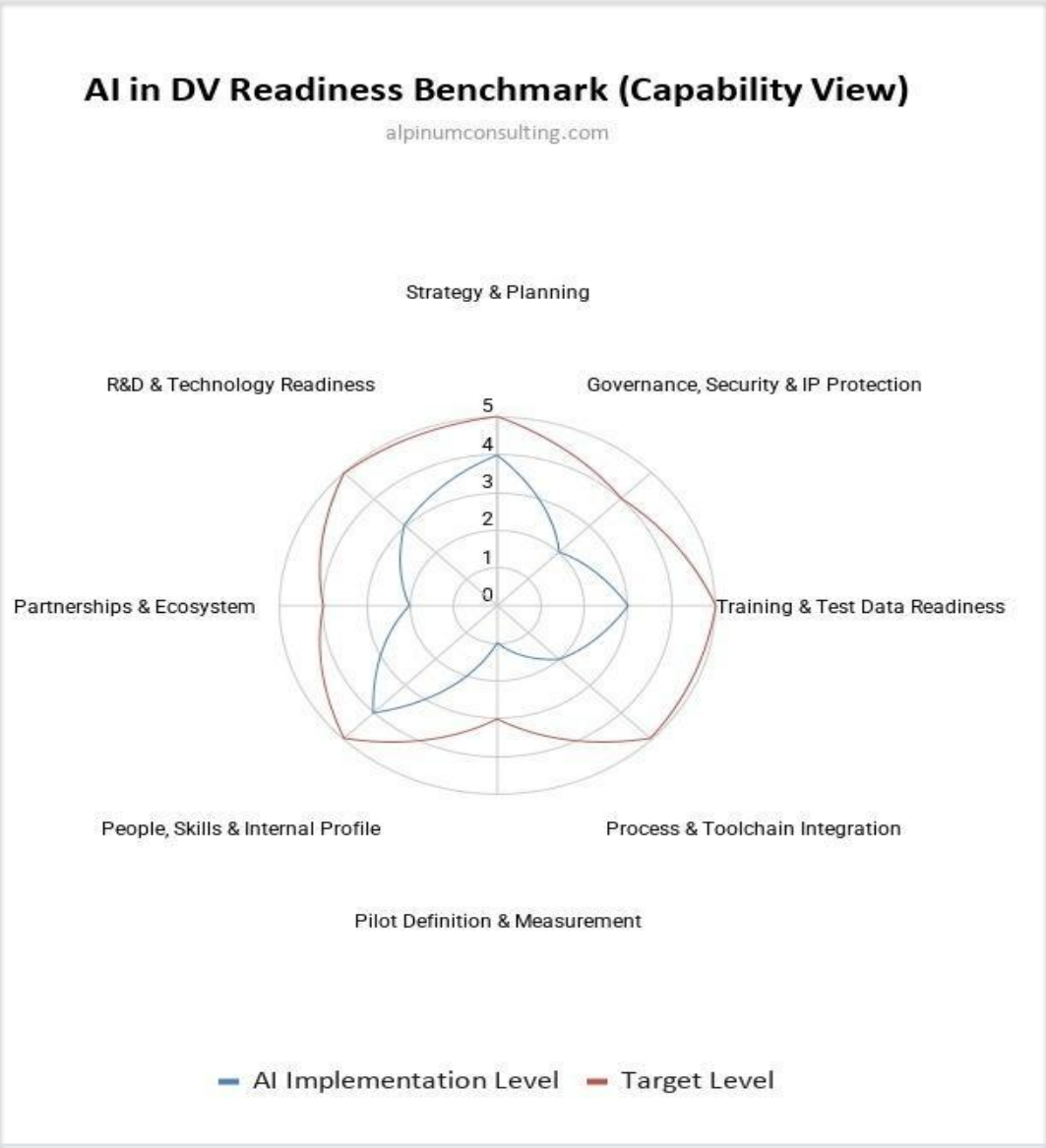
Why it matters

Assumptions become timing-sensitive, analogue-aware or dependent on interactions outside the training distribution.

Implication: keep first pilots bounded, repeatable and close to a known review baseline.

AI in DV Capability Maturity Model

Example capability assessment framework for AI-assisted design verification adoption



Key capability areas

- **Strategy & Planning -** Defines AI adoption goals aligned to measurable DV and programme outcomes.
- **Governance, Security & IP Protection -** Maintains traceability, review control, security boundaries, and IP protection.
- **Training & Test Data Readiness -** Requires high-quality training and test data, structured logs, reusable datasets, metadata discipline, and readiness for AI-assisted processing.
- **Process & Toolchain Integration -** AI must integrate cleanly with UVM, formal, regression, CI, and debug workflows.
- **Pilot Definition & Measurement -** Pilots need bounded scope, measurable baselines, and predefined success criteria.
- **People, Skills & Internal Profile -** Engineers remain accountable for semantic review, approval, and sign-off decisions.
- **Partnerships & Ecosystem -** Adoption may depend on EDA vendors, infrastructure partners, internal support, and external expertise.
- **R&D & Technology Readiness -** The organisation must evaluate model maturity, infrastructure capability, and operational scalability.

A safe pilot starts with a bounded engineering problem

1	**Scope**	Select one stable block, protocol or regression problem.
2	**Baseline**	Measure the current manual process before applying AI.
3	**Data readiness**	Normalise logs, metadata, revisions and bug states.
4	**Human boundary**	Define what AI may suggest and what engineers must approve.
5	**Scorecard**	Track productivity and assurance separately.

Pilot scorecard: measure productivity and assurance separately

Productivity signal	Assurance control	Acceptance gate
• Time saved in triage • Reduced duplicate analysis • Fewer wasted regression runs • Faster repeated-failure grouping	• Traceable inputs • Version-aware datasets • Reviewed artefacts • No unapproved waivers	• Predefined success criteria • Baseline comparison • Review acceptance rate • Stop / continue decision

Never report raw coverage uplift as success unless reachability, semantic value and review quality are also checked.

Governance rules for “AI in DV” adoption

A practical policy can be simple: AI may draft, rank and summarise. Engineers approve.

Allowed with review	Needs strict controls	Do not automate blindly
• Failure clustering	• Generated testbench code	• Coverage waivers
• Log summarisation	• Formal support artefacts	• Property approval
• Candidate assertions	• Training on internal data	• Sign-off judgement
• Coverage-gap suggestions	• Regression prioritisation	• Release recommendation

Minimum evidence pack: source data, prompt/configuration, generated output, engineer review result, decision log and known limitations.

Training exercise: choose the first safe pilot

In groups, select one proposed use case and test it against the qualification checklist.

Regression triage for a stable block

Assertion suggestion for known protocol

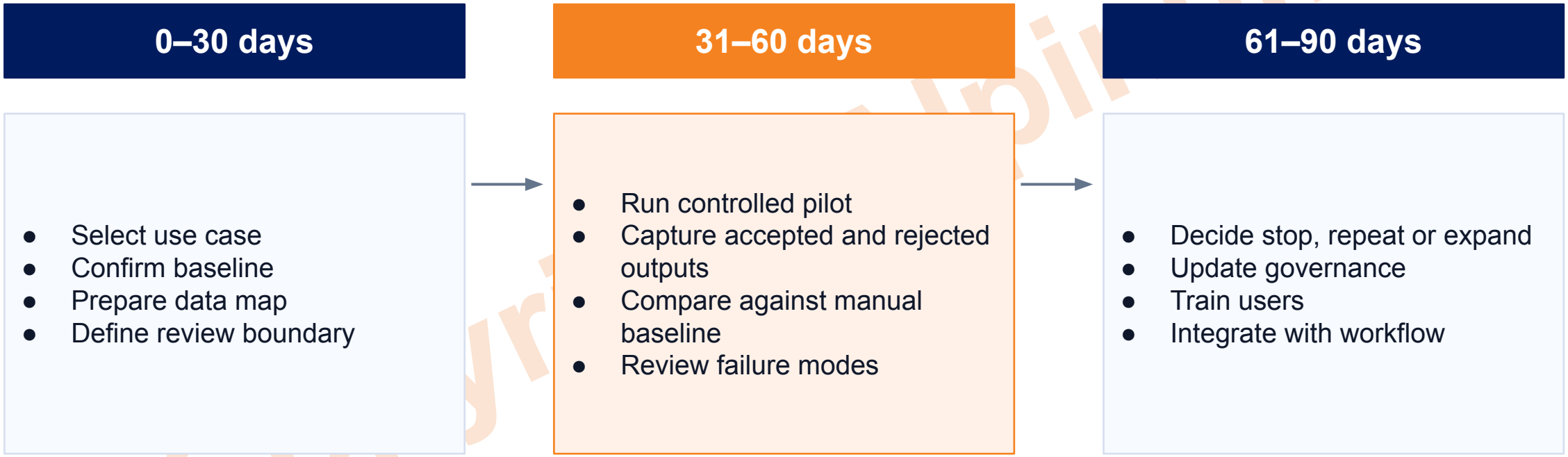
Coverage-gap ranking in mature environment

Failure clustering for noisy regression farm

Answer these five questions

What is the bounded problem?
What current baseline will we compare against?
What data is needed and is it traceable?
What can AI suggest and what must engineers approve?
Which metric decides whether to expand or stop?

A phased route from experiment to capability



Aim: move from isolated AI experiments to measurable DV capability, without weakening engineering accountability.

Key takeaways

1. AI helps most when the task is bounded, repetitive, data-rich and reviewable.
2. AI hurts when teams confuse generated artefacts or dashboards with verification evidence.
3. A safe pilot needs scope, baseline, data discipline, human review boundaries and a scorecard.
4. The objective is not full automation. It is better prioritisation, faster learning and stronger decision confidence.

Questions and discussion

References: Accellera UVM Working Group; Bennett and Eder 2025; Ye et al. 2025; Menon et al. 2025; Pothireddypalli et al. 2026; Accellera UVM-MS 2025.