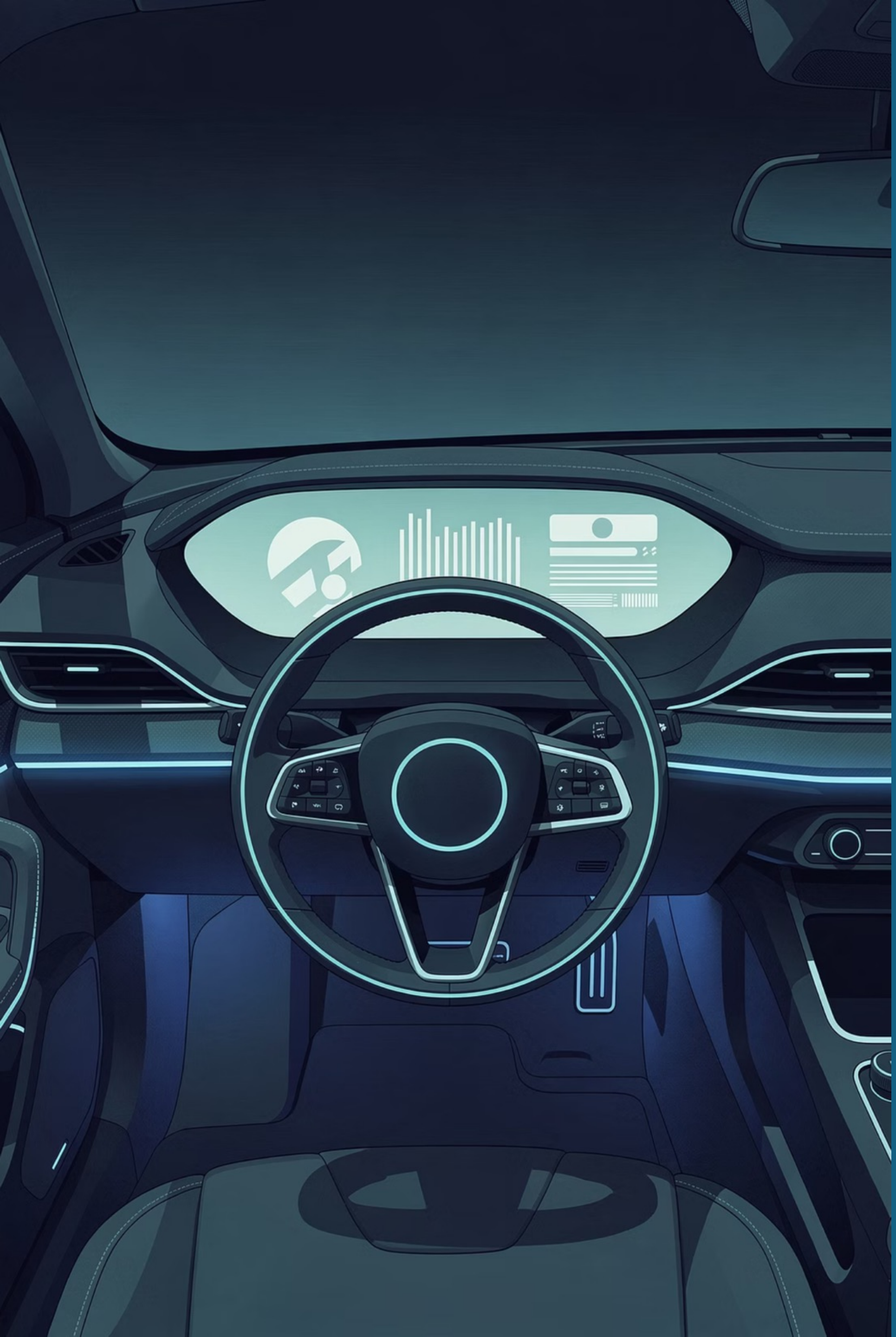


AI-Powered Product Development

Building better automotive products with smaller, faster, cross-functional teams

90-MINUTE HANDS-ON WORKSHOP · INDUSTRIAL DEVOPS NOW

Dr. Robin Yeman
Technical Fellow
Leidos

Why We're Here

Today, building that one feature takes many teams and many months. This workshop asks: what if it didn't have to?

AI IMPACTS ACROSS SDLC

Planning

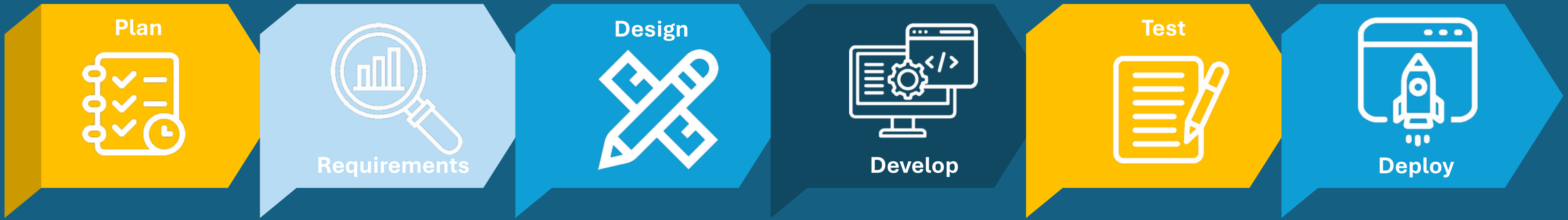
- Requirement analysis
- Generate User Stories
- Requirements Traceability
- Develop Roadmaps
- Dynamically adapt based on changing constraints.

Arch / Design

- Architecture writing / analysis
- Create API
- Sequence flow diagrams
- Generate Data Models
- UI Assist (Wire frame, persona, story maps)
- Simulate / Validate Choices
- Integrate Compliance Constraints

Test

- Test Case Optimization
- Unit /Integration/Regression Generation
- Simulate multiple scenarios
- Triage Failure
- Open bugs, generate rollback / fix



Requirements

- Summarize stakeholder needs
- Sentiment Analysis
- Create Functional Requirements
- Discover new requirements
- Update Backlogs

Develop

- Code Generation / Debug
- Automated Code Review
- Code explain / commenting
- Refactoring
- Code translation
- Autonomous code agents

Deployment

- Optimize release timing
- CI/CD Pipelines
- Infrastructure as code
- Monitor pipeline
- Dynamically reconfigure resources
- Auto fix failed builds

Workshop Objectives

Everything we do today ladders up to four outcomes.

Smaller Teams

Cross-functional product pods, not siloed departments

Faster Iteration

Hours to working artifacts, not weeks of handoffs

Better Traceability

One auditable thread from need to deployment

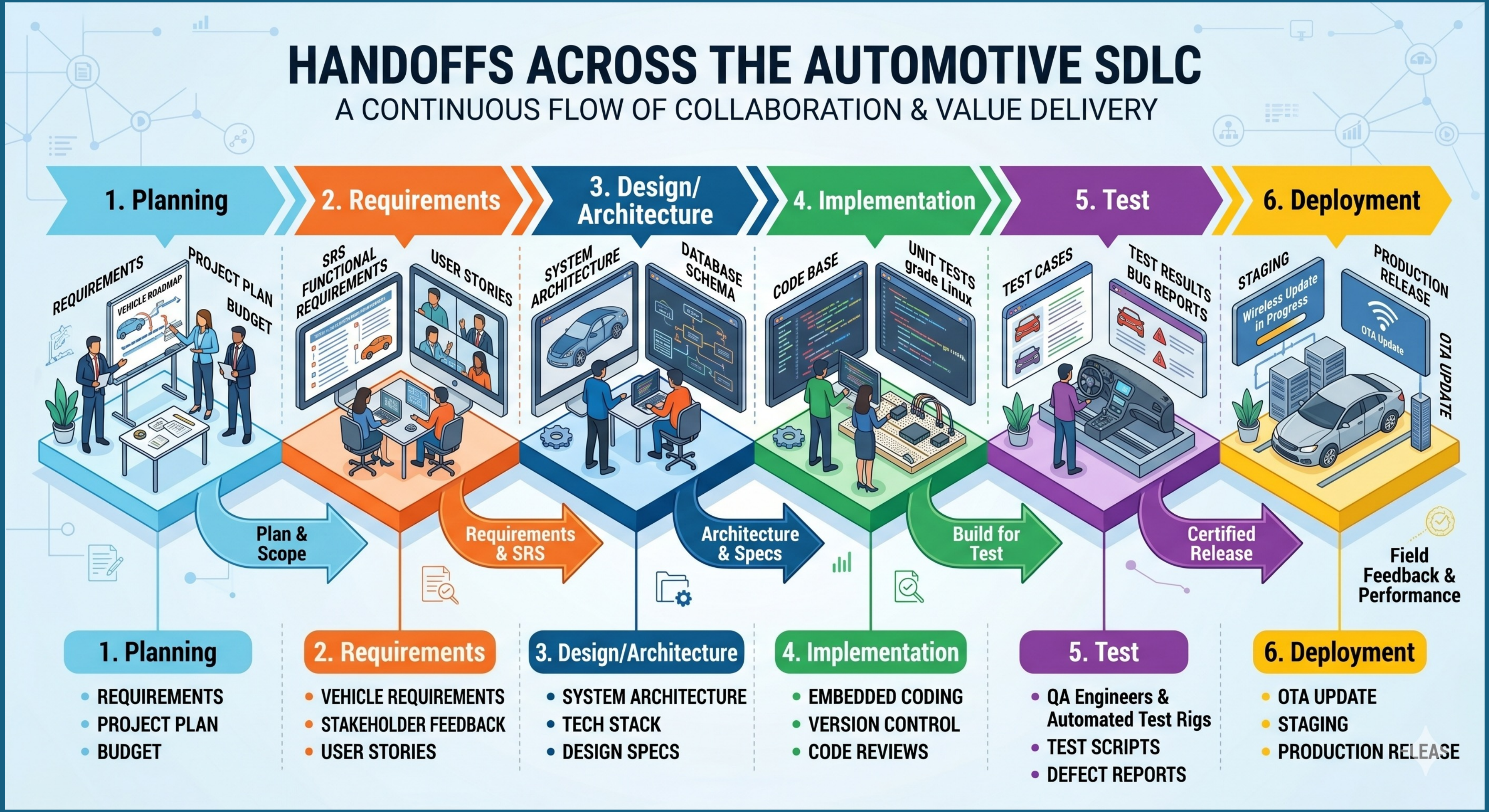
Outcome-Focused Delivery

Ship features that move the needle for real people

Automotive Development Today

Specialized functions create handoffs, delays, and rework. Each team is excellent at what it does – but the seams between them carry enormous hidden cost.

The Cost of Handoffs



Each handoff is a delay. Specialization is valuable – but sequential handoffs no longer scale alone.

Why Specialization Exists

Specialization isn't wrong – it's essential. Risk reduction, regulatory compliance, and deep domain expertise are non-negotiable in automotive development.

The challenge isn't expertise. It's the **sequential, siloed structure** that forces every idea through a long chain of handoffs before it reaches a customer.

The goal is not to eliminate specialists – it's to change when and how they engage.

AI as Capability Amplification

AI makes expertise available at the point of need. It augments specialists – it does not replace them. A systems engineer with an AI copilot can think at the speed of the architecture, not the speed of the document.

The Future-State Team

One small, cross-functional product team – with AI as a permanent, always-on team member.

Product Owner

Owns the outcome. Defines what success looks like for the driver and the business.

Scrum Master

Coach guides the team, optimizes delivery

Team

Cross-functional. Requirements, architecture, code, and test – in one pod.

AI Copilot

Generates artifacts, traces requirements, flags gaps, and accelerates every step.

Driver Monitoring System

Detect drowsy, distracted, or medical-emergency states – and respond safely. **This is our scenario for the entire live build.**

The Customer Need

Protect a tired or distracted driver – **without constant false alarms**.

That tension is real and consequential. A system that cries wolf too often gets ignored. A system that misses a genuine emergency can cost a life.

Everything we build today starts here with a human need, not a technical spec.

1 Planning

XXXXXX

1 2 3 4 5 6

➢ Prompt 1

You are a cross-functional automotive product team. Analyze the Driver Monitoring System concept. And generate Stakeholders, Business Objectives, Customer Needs, Risks, Assumptions, Success Metrics, Prioritized Outcomes

Whats the Plan

② Requirements

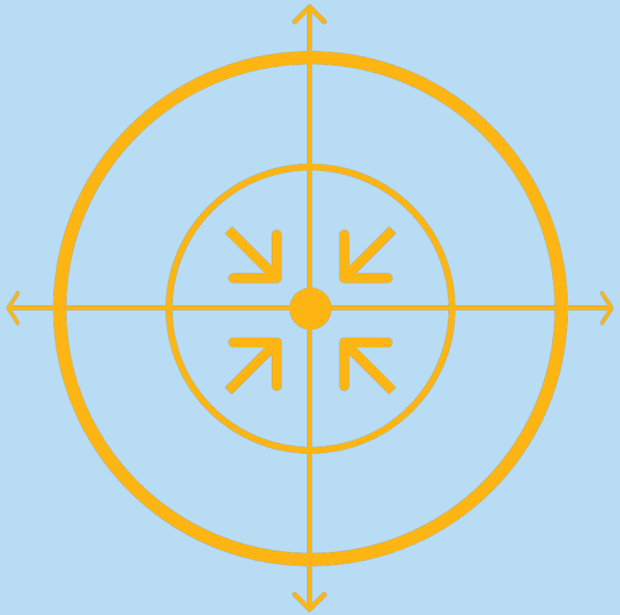
We begin where every program begins – but we collapse weeks of elicitation into minutes.

1 2 3 4 5 6

➢ **Prompt 2**

You are a senior automotive systems engineer. Using the Planning outputs (stakeholders, customer needs, business objectives, success metrics, prioritized outcomes), produce a requirements specification for the Driver Monitoring System (DMS): functional requirements · non-functional requirements (latency, accuracy, privacy) · constraints & assumptions · acceptance criteria. Trace each requirement back to a stakeholder need or business objective from Planning

What's missing? What would you challenge?



What Good Requirements Cover

- **Stakeholder needs** – driver safety, OEM liability, regulatory bodies
- **Functional** – detect drowsy, distracted, medical-emergency states
- **Non-functional** – latency under 200 ms, accuracy ≥ 95%, on-device privacy
- **Acceptance criteria** – objective, measurable pass/fail conditions

Traceability starts here. Every downstream artifact must link back to a requirement.

3 Architecture / Design

From approved requirements to a system architecture – same session, same team, no handoff.

1 2 3 4 5 6

- **Prompt 3**

Using the approved requirements, create a concise architecture description:
System context diagram (in text) · Major components: sensors, perception/AI, decision logic, HMI, vehicle interface · Interfaces & data flows · Failure modes · Safety considerations · Privacy & cybersecurity considerations

Where could this go wrong, and how does it stay safe?

DMS Architecture at a Glance

Sensors

In-cabin camera, IR illuminator, optional steering-torque and eye-tracking sensors

Perception / AI

On-device ML model classifying driver state with confidence scoring

Decision Logic

Threshold engine determining alert level: advisory, active, emergency

HMI / Vehicle Interface

Haptic, audio, and visual alerts; vehicle braking and lane-keep integration

Model The System

A structured, machine-readable representation of the design – traceable to every requirement.

➢ **Prompt 4**

Using the architecture, generate a simple SysML v2 textual model:

· Requirements · Actions / functions (DetectDrowsiness, AssessConfidence, TriggerResponse) · Interfaces / flows · Major system elements

Focus on readability, not tool-specific correctness.

What should we watch for?

Model: What It Captures

Requirements

Formal, numbered, with satisfaction links

Actions

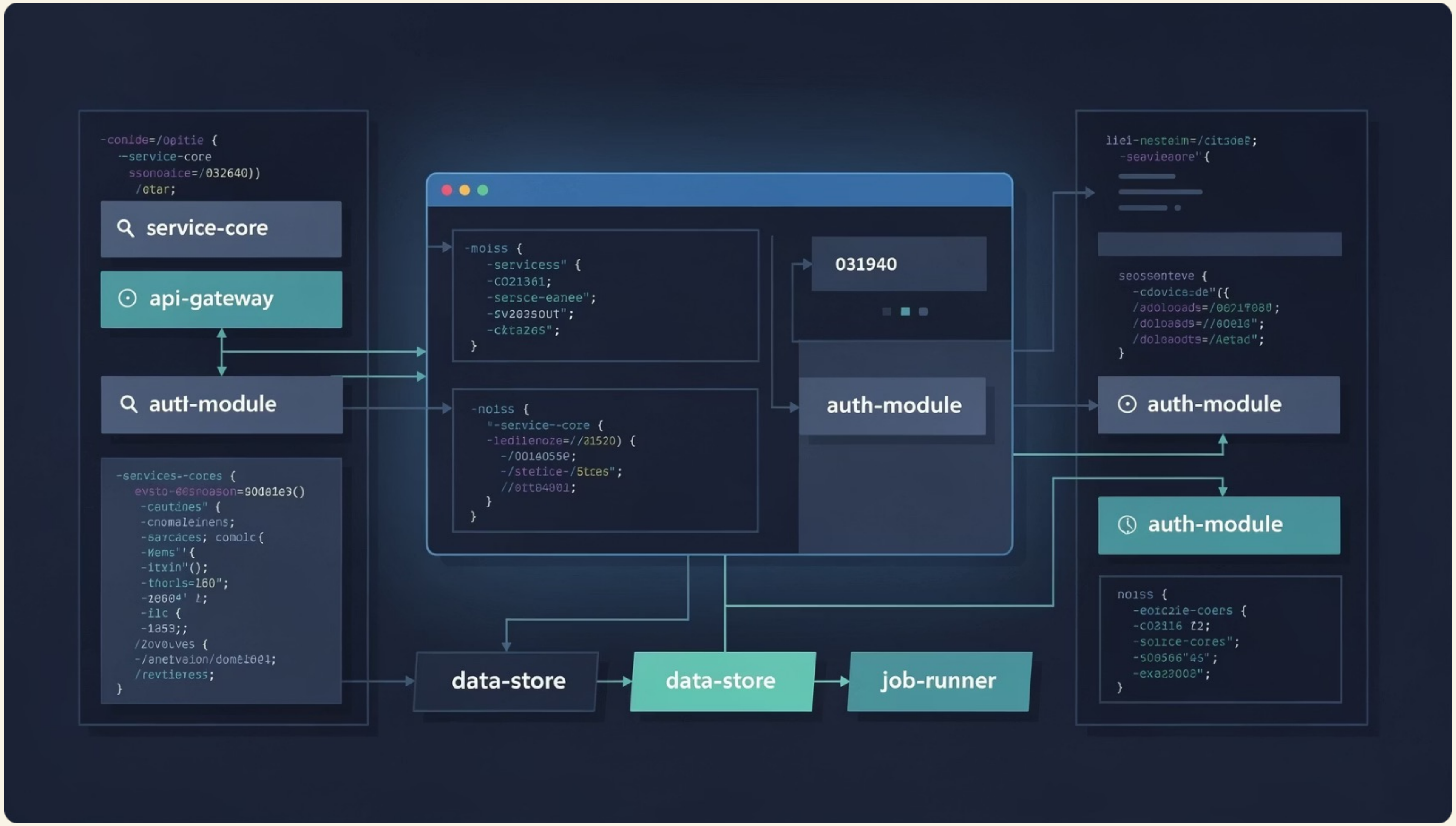
DetectDrowsiness → AssessConfidence → TriggerResponse

Interfaces

Data flows between sensor, model, logic, and HMI

System Elements

Named components with allocated responsibilities



④ Implementation

From design to logic – no separate dev-team handoff. The same team that wrote the requirements generates the implementation approach.

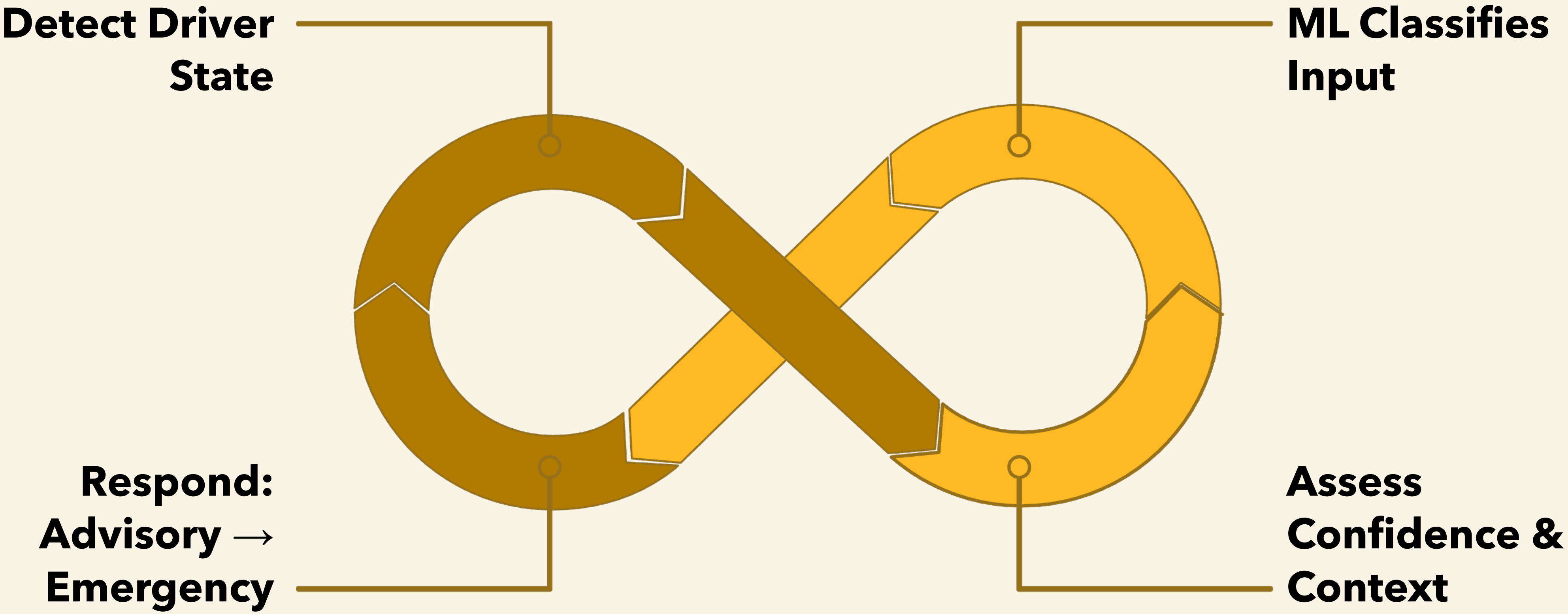
① ② ③ ④ ⑤ ⑥

➤ **Prompt 5**

Using the architecture and the SysML model, generate: high-level implementation approach · pseudocode for the detect-assess-respond loop · logic flow & data processing steps · assumptions. Explain how the solution satisfies the requirements.

Idea to logic with no separate dev-team handoff.

The Detect → Assess → Respond Loop



This core loop is the heartbeat of the DMS. Every requirement, every test case, and every architecture decision traces back to these three actions.

⑤ Verification

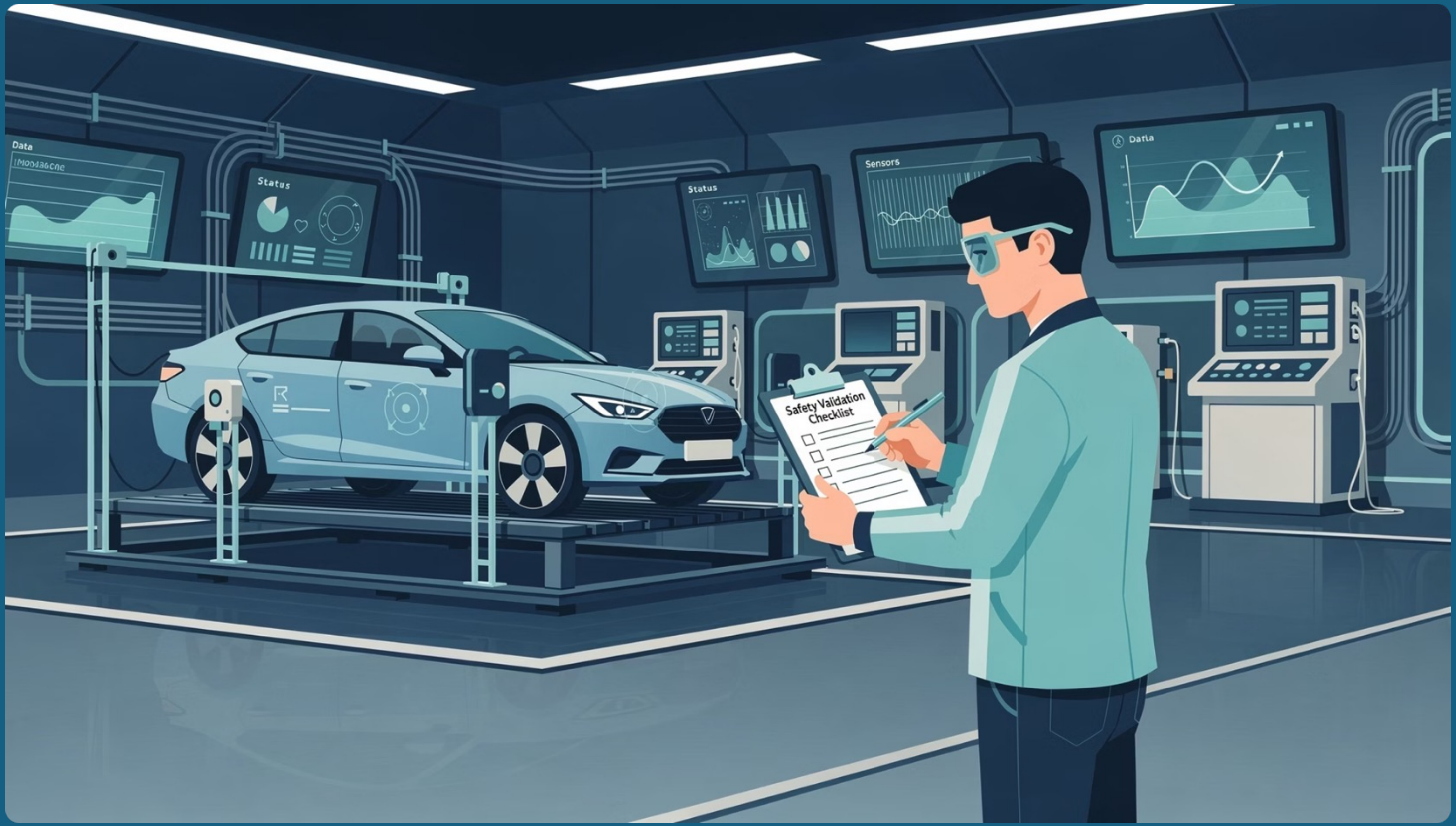
AI reviews its own implementation against requirements – flagging gaps before a human engineer makes the final call.

1 2 3 4 5 6

- **Prompt 6**

Generate test cases from the acceptance criteria, then review the proposed implementation. Identify: which requirements are satisfied · which test cases pass · potential defects · risks (false alarms, missed detections) · recommended improvements. Provide a pass/fail assessment

Where does human expertise have to make the final call?



What Verification Surfaces

- **Requirements coverage** – which reqs are satisfied, which are not
- **Test readiness** – which test cases would pass in the current design
- **Defect candidates** – logic gaps, unhandled states, race conditions
- **Risk flags** – false-alarm rate, missed-detection scenarios

⚠ AI can identify. Only a qualified engineer can accept the risk and sign off.

6 Deployment

OTA plan, rollback strategy, monitoring metrics, and success criteria – the final artifact in the digital thread.

1 2 3 4 5 6

- **Prompt 7**

Using the verified implementation, create the deployment plan: OTA deployment plan & rollback strategy · monitoring metrics (accuracy, false-alarm rate, driver trust) **— mapped back to the Success Metrics from Planning** · release notes · customer impact assessment · success criteria after deployment

We just reached a deployment plan – how many teams would this normally take?

Deployment Essentials

1

OTA Rollout

Staged deployment: 1% → 10% → 100% fleet, gated on real-world accuracy metrics

2

Rollback Strategy

Automatic revert if false-alarm rate exceeds threshold within 48 hours of release

3

Live Monitoring

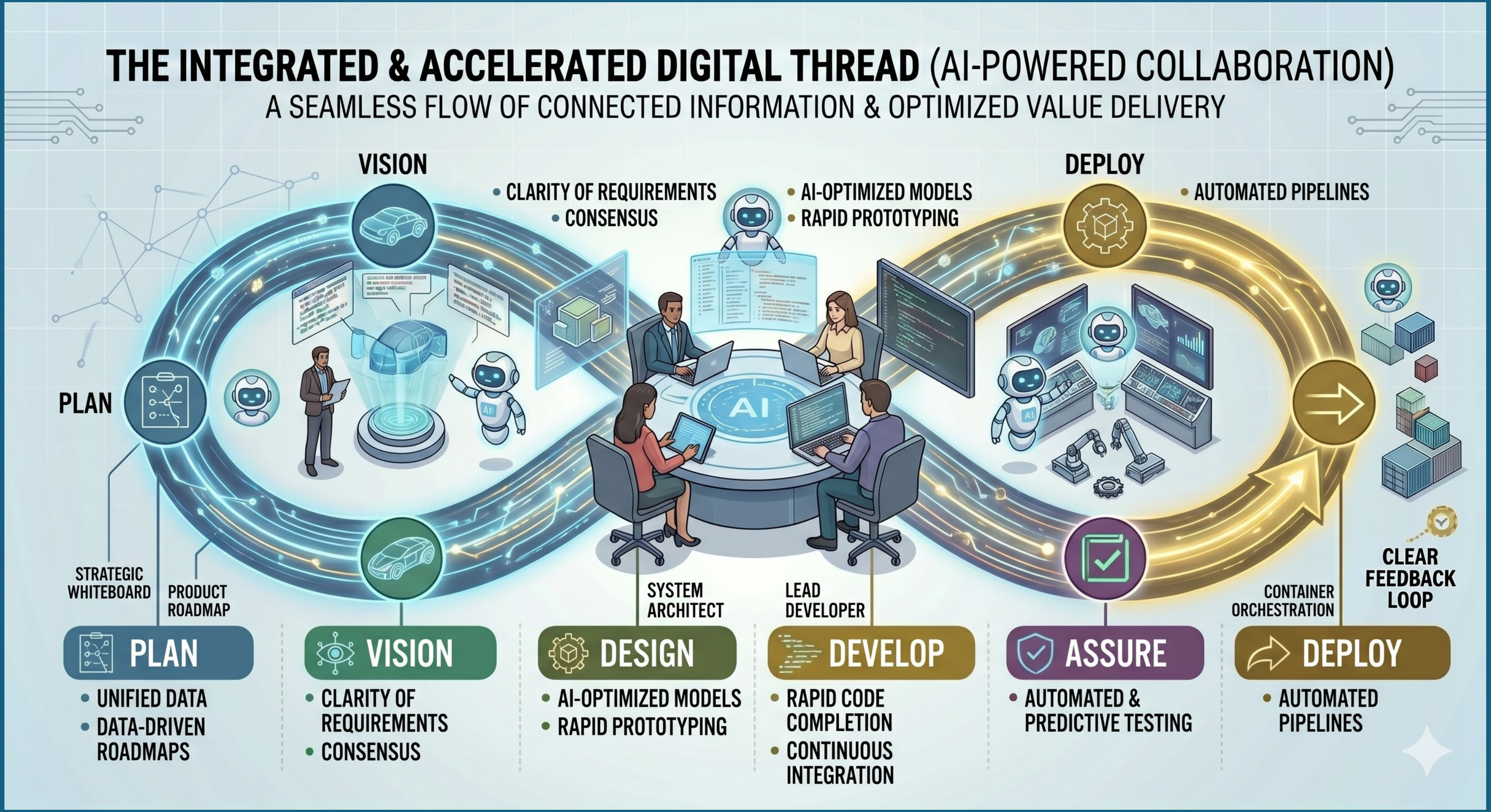
Track: classification accuracy, false-positive rate, driver override frequency, trust score

4

Success Criteria

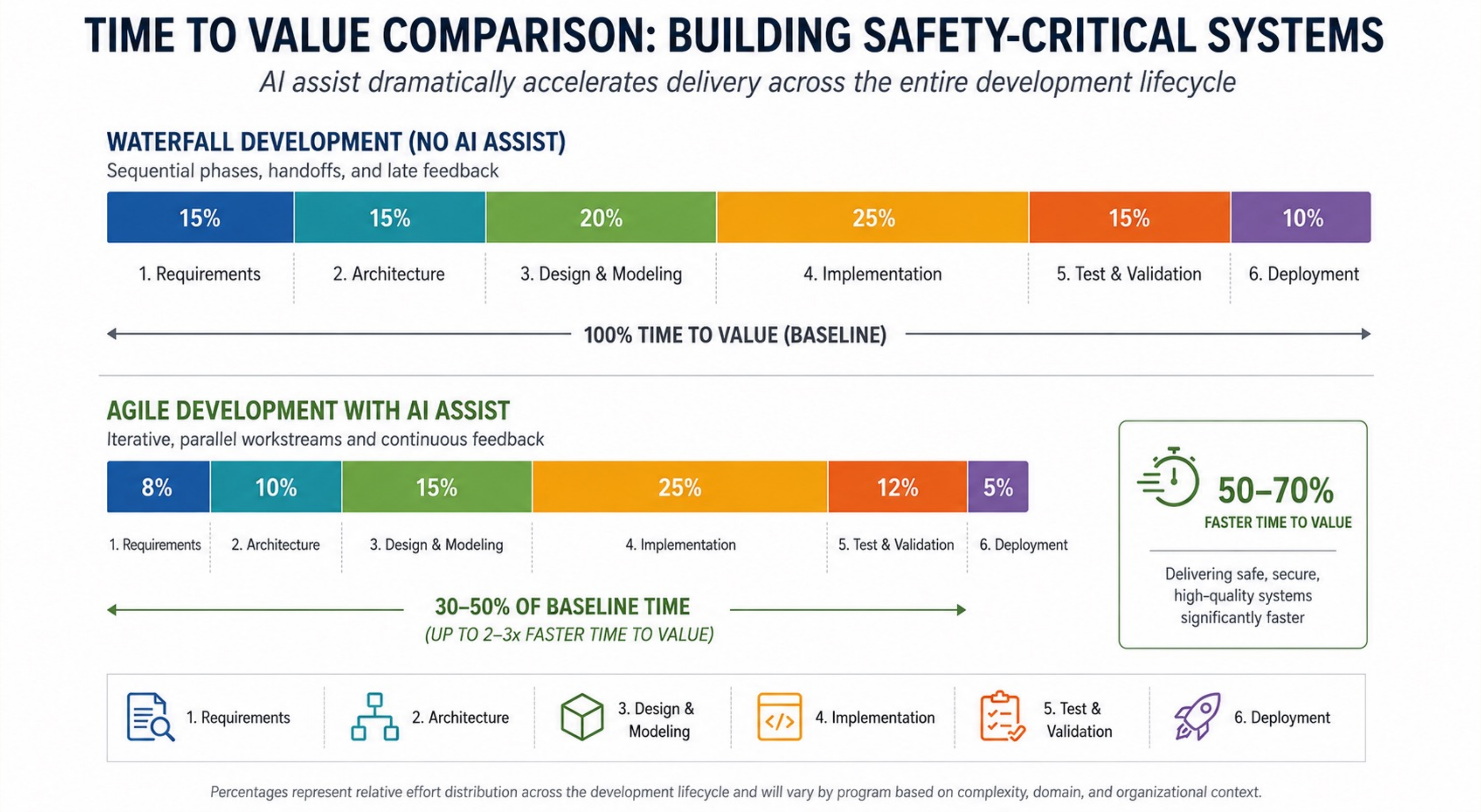
≥95% accuracy, <2% false-alarm rate, zero critical missed-detection events at 30 days

The Digital Thread



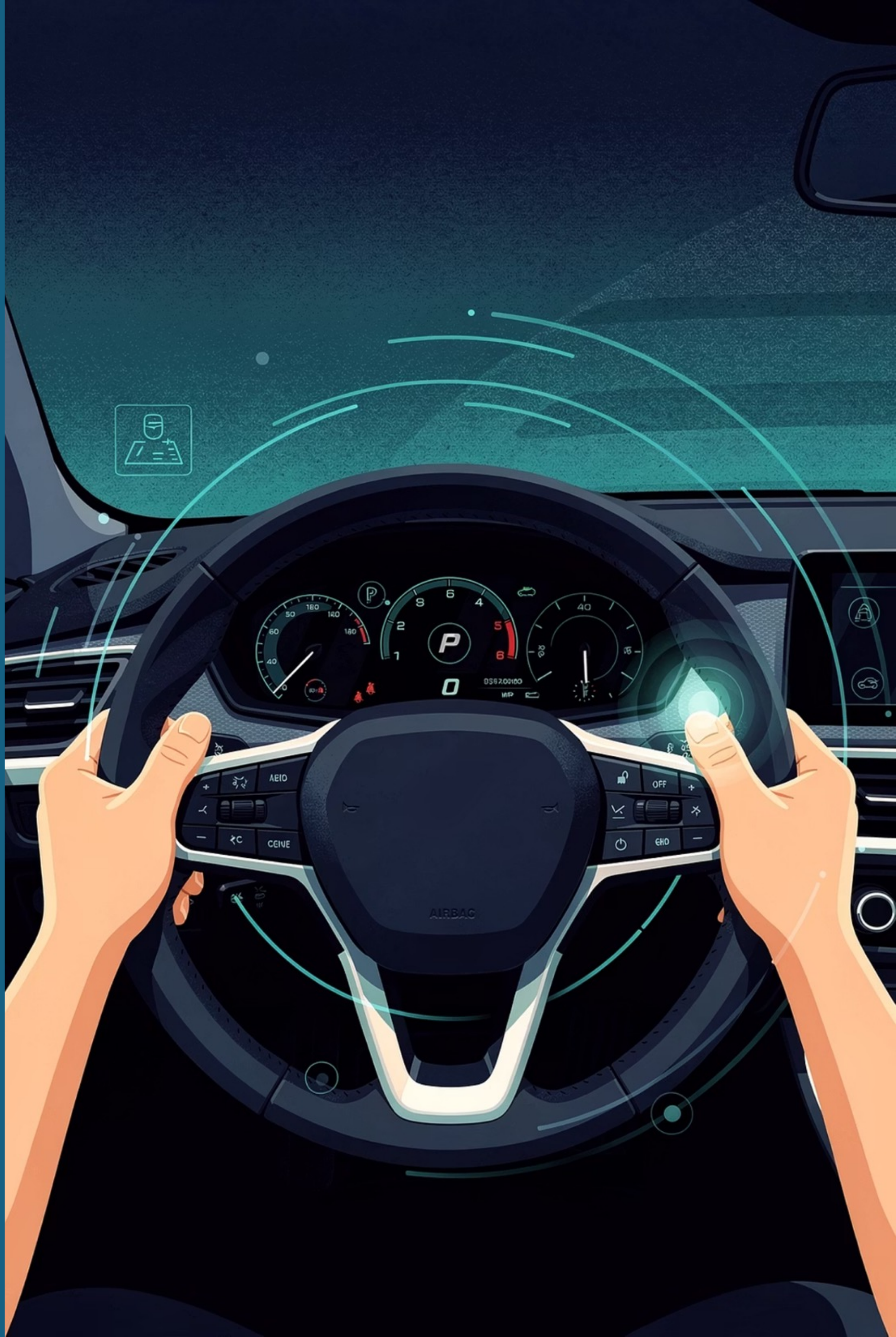
In a safety-critical system, that traceability isn't a nice-to-have – it's the evidence that the product was built right. Scroll back through the live chat: every artifact links to the one before it.

Traditional vs. AI-Enabled Timeline



Where Humans Stayed in the Loop

AI assisted. People decided. Every step of the build included a moment where human judgment – not the model – determined what was acceptable.



The Human Decisions That Matter

Confidence Thresholds

What confidence level is high enough to alert – and who sets that bar?

Safety Aggressiveness

How forceful should the intervention be? That is a values decision.

Bias Checks

Does the model perform equally across driver demographics and lighting conditions?

Accountability

When something goes wrong, a person – not a model – answers for it.

Governance

Regulatory compliance, functional safety standards, and corporate policy cannot be delegated to AI.

Expert Reviews

Domain specialists validate AI outputs – the review role is more important, not less.

What This Means for Your Organization

1 Today

Sequential handoffs. 6+ specialized teams. Months from idea to deployment plan.

2 Transition

Pilots with small cross-functional pods. AI copilots in the workflow. Specialists as reviewers.

3 Future

Outcome-oriented product teams. AI-generated artifacts. Specialists governing, not executing sequentially.

Three Things to Take With You

Smaller Teams Win

Cross-functional pods move faster and maintain better traceability than sequential specialist chains

Faster Learning

AI compresses the feedback loop – ideas become artifacts in hours, enabling earlier validation

Better Outcomes

Same rigor, same safety standards – built by a team focused on a human need, not a handoff schedule

Humans remain accountable throughout. AI accelerates the work. People own the decisions.

Questions & Discussion

Where would you pilot AI first?

We just took a feature that protects a tired driver – from idea to deployment plan – as one small team, in minutes. **Now imagine this across your entire portfolio.**