

eNVD Agentic Delivery: sprint value and the case to extend

Prepared for Integrity Systems / Meat & Livestock Australia by Terem Technologies • eNVD stream • Sprint 1, 15 June – 10 July 2026

1. Executive summary

The four-week agentic sprint set out to change how the eNVD team builds software, not to move a single dashboard number. It has done that in engineering, lifting the eNVD pod from ad hoc AI use (Level 0) to a governed, single-agent workflow (Level 1). The full pipeline - skills, MCPs, hooks, evals and guardrails - is in place, three live features have moved through it end-to-end, and the technical friction that used to gate work is largely gone. The team is measurably more capable than at any point before.

The value shows up as time returned on every piece of work. On current evidence the team is on track to save around three hours per ticket in engineering, where the tooling is implemented and in daily use. The same saving, applied across the three roles that touch a ticket - business analysis, engineering and quality assurance - would be roughly nine hours, close to a full working day per ticket. That fuller figure is a projection: engineering is being demonstrated now, and business analysis and QA would follow by extending the approach. Even at the engineering-only level it points to a large amount of capacity handed back to the business each year.

Two points are worth drawing out. **First, the value is capacity and speed, not cost saving.** MLA is funded to deliver this work, so the question is not how many tokens are consumed, it is how much more the team can ship and how quickly. Second, the sprint has so far only accelerated the engineering part of the flow. The largest remaining gains sit upstream in business analysis and downstream in QA, which the next phase would address.

The shape of the return

A \$66k sprint over four weeks. Engineering savings are being demonstrated now; extended across business analysis and QA, the projected return is in the order of 600 working days of capacity a year for a single pod. **The same pattern repeats for every pod it reaches.**

2. What we implemented this sprint

The sprint focused on engineering, where agentic delivery is now embedded and being used to build live features. Business analysis and QA have been brought in through enablement rather than full implementation. That is deliberate: engineering is where the model is being proven before it is widened. Nine of the nine sprint deliverables are complete or on track for Friday's close.

- **Engineering, implemented.** The eNVD pod moved from ad hoc AI use (Level 0) to a governed, single-agent workflow (Level 1). The full pipeline - skills, MCPs, hooks, evals and guardrails - is in place, and three live features have moved through it end-to-end, closing this Friday.
- **Cost and usage, in place.** Usage and cost are tracked through Claude's built-in analytics, which proved sufficient, so no custom dashboard was needed. The team will self-service reporting from there.
- **BA and QA, enablement started.** Playbooks and role guides have been shared with the BA and QA leads for feedback, and the team has begun tightening how tickets are made ready for

development. Full agentic BA and QA delivery is scoped as the next phase, not yet implemented.

So the time savings in this document are being demonstrated in engineering now. The equivalent savings in business analysis and QA are projected, on the same basis, and would be realised by extending the approach into those roles.

3. How we are measuring value

Cycle time is the easy measure - a ticket starts on Monday and finishes on Friday, so five days. It is also misleading. It hides the back-and-forth that happens when a ticket is not fully ready, and it does not capture lead time, which is where much of the real delay sits. Measuring the sprint on cycle time alone would understate the value and point at the wrong place for the next improvement.

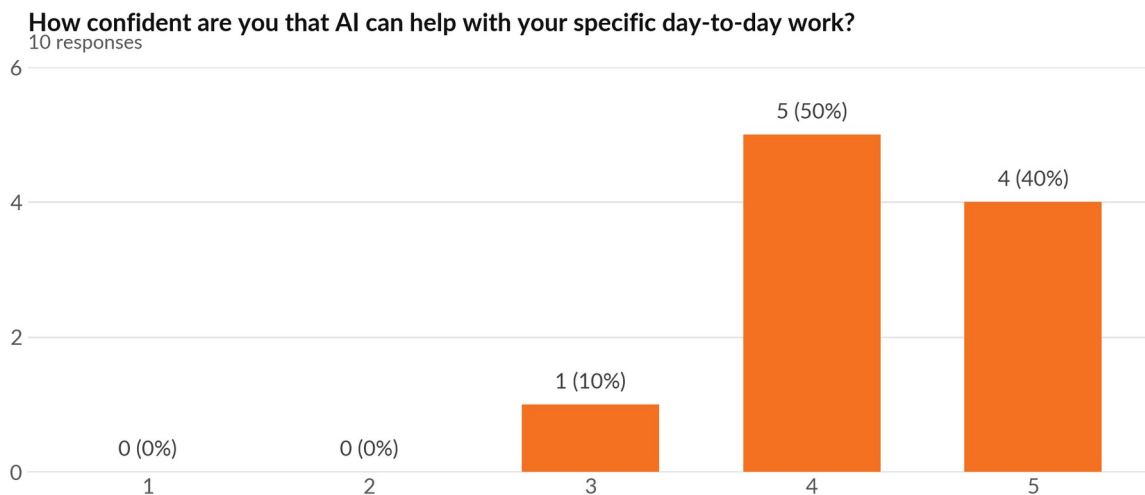
So rather than argue a percentage against an unreliable baseline, the value case rests on two things the team has agreed to track.

- **An 80% adoption target** across everyone involved in the engagement, including business analysis and QA, gathered through a short survey. This tells us whether the change has actually taken hold, not just whether the tooling is available.
- **Around three hours saved per ticket, per role**, captured with the delivery team directly, so the numbers are grounded in the work and owned by your team.

Usage and cost are tracked alongside these through the built-in analytics, but they are context, not the headline. The point of the engagement is the value gained in the delivery flow, not the price of the tooling.

4. Early signal from the team

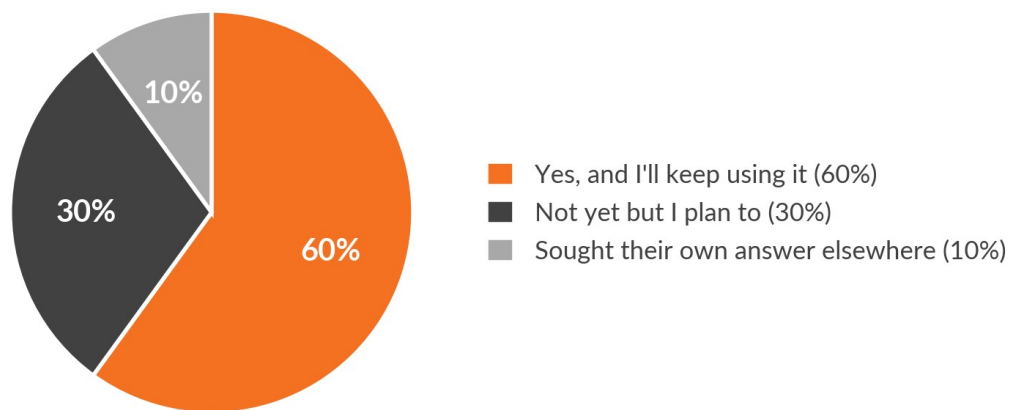
A short pulse survey after the AI training offsite gives an early read on how the team feels. It is a small sample of ten responses, so it is a signal rather than a settled measure, but the direction is clear. The sprint's own adoption survey will formalise this across the whole team.



Confidence is high: nine in ten respondents rated their confidence that AI can help with their day-to-day work at 4 or 5 out of 5, and none rated it below the mid-point.

Did you try something new with AI at work this week based on the workshop?

10 responses



Adoption has already started: sixty per cent had tried something new with AI that week and intend to keep using it, with a further thirty per cent planning to.

The one respondent who looked elsewhere wanted answers to role-specific questions a general offsite could not cover. That is exactly the gap the next phase closes, by embedding agentic support in the specific work of business analysis and QA rather than teaching AI in the abstract.

The sprint retro reinforced the same picture. The team backed the model and valued how directly AI plugged into their real codebase, but they were clear that the hard part is still requirements refinement, and that the tooling only performs when the inputs - skills, context and ready tickets - are right. Both BA and QA said they want more of this way of working, not just engineering.

5. The value, quantified

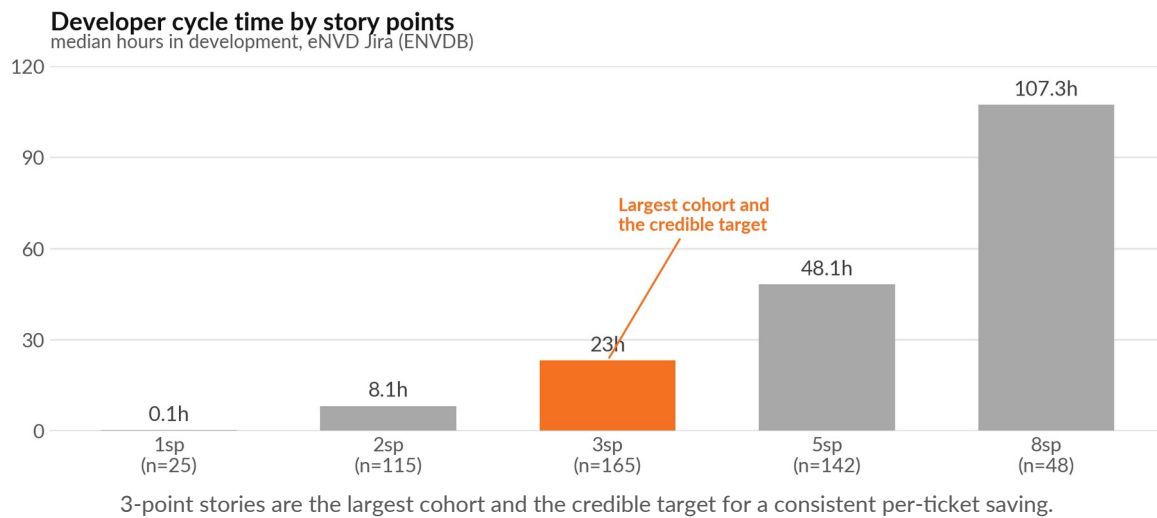
The figures below are planning assumptions drawn from the sprint and the delivery team's own estimates. The engineering component is being demonstrated now; the business analysis and QA components are projected on the same basis and would be realised by extending the approach. They are deliberately conservative and are there to be confirmed with your team against MLA's own data, and the conclusion holds even if the inputs are halved.

Measure	Indicative value
Time saved per ticket, per role	~3 hours
Roles across the pod	BA • Engineering • QA
Time saved per ticket, whole pod	~9 hours (about one day)
Annual ticket volume, per pod (indicative)	circa 600 tickets
Time recovered per year	~600 working days
Equivalent capacity, per pod	~3 full roles
Sprint investment	\$66k over four weeks
Adoption target	80% of the team

Read the table as capacity rather than cost. Around 600 working days returned to a single pod, against a sprint investment of \$66k, is a substantial return. Because a pod is one BA, one engineer and one QA, the saving scales with the number of pods the approach is applied to.

6. What your delivery data confirms

To pressure-test the value case we analysed 1,437 issues from your eNVD Jira, the ENVDB project. Cycle time is a blunt measure of success, but as a map of where time accumulates it is very useful, and this data confirms both the shape of the opportunity and where the largest gains sit.



Looking at the engineering-owned states, the typical ticket moves quickly but a minority stall for far longer:

Measure	In Development	Dev + Code Review
Median	23.4h (~3 days)	42.5h (~5 days)
P75	96.6h (~12 days)	127.1h (~16 days)

The gap between the median (about three days) and the P75 (about twelve) is the long tail: a minority of tickets sit far longer than the typical one, usually waiting on clarification or caught in rework rather than in active coding. That tail is where the biggest gains are, and it points upstream to how work is specified.

One important distinction. These are cycle-time figures, the calendar time a ticket spends in a status, not hours actively worked. For a mature team, active effort is usually only a third to a half of that. The three-hour target is therefore a reduction in cycle time, which light effort-logging on a sample of tickets during the sprint will confirm precisely.

The value case is modelled on circa 600 tickets per pod per year, with validation continuing. Recent throughput sits in the same range: the eNVD project completed 483 tickets in FY26. At three hours saved per ticket, that is engineering-only capacity of roughly 180 to 225 working days a year, close to one developer freed, before business analysis and QA are added. Extending the saving across all three roles reaches the 600 working days, or three roles, in the value table.

7. Where the delivery flow still leaks

The sprint accelerated the middle of the flow. Speed gained in the middle is capped by a slow start and a slow finish, and that is exactly where the remaining friction now shows up.

- **Work arriving before it is ready.** A meaningful share of tickets reach the build stage less than 75% ready, which forces analysts and engineers back into rework mid-sprint. The retro named requirements refinement as the hard part that remains, and noted the tooling only performs when the inputs, skills, context and ready tickets, are right. This is the clearest near-term opportunity and it sits with business analysis.
- **Testing is the emerging bottleneck.** Now that code generation is fast, testing is becoming the constraint. The team has already raised the obvious question: if developers are this quick, how

does QA keep pace across every use case? Speed gained upstream is lost if validation cannot match it.

- **The value is contained to one pod.** The gains proven so far are real but contained to the pod that has them. They stay locked at one pod until the pattern is extended.

8. What comes next: extend to business analysis and QA

The next phase completes the flow rather than repeating the same work on more people. The retro was explicit that four weeks was not enough to fully test and extend the model, and that both BA and QA want more of this way of working. The aim is for a piece of work to move from idea to live with agentic support at every step, so the engineering speed already proven is no longer held back by what happens before and after it.

- **Business analysis:** Backlog items arrive genuinely ready to build. Definition to deploy shortens at the source, not just in the build.
- **Quality assurance:** Automated coverage and faster validation keep pace with faster delivery, so speed holds through to done and quality is not traded away.

Your delivery data shows exactly where wider agentic use in business analysis and QA would move the numbers:

Signal in your data	What is driving it	What BA and QA agentic uplift changes
Long cycle-time tail (P75 ~12 days vs a ~3-day median)	Tickets stalling on clarification and mid-build rework when specs are not dev-ready	BA agentic produces dev-ready specs, rules and edge cases up front, shrinking the tail
Code review adds ~19 hours to the median	Manual pull-request review load	Agentic review and test generation compress review turnaround
Spikes run ~42 hours at the median	Architectural lookup, API-doc reading and prototyping done by hand	Agentic research and prototype generation compress investigation time
Forms (111) and UI/UX (112) tickets	Repetitive component scaffolding and boilerplate	Code generation gives near-immediate cycle-time reductions
Consignment Creation is the largest area (210 tickets)	Structured, repeatable form logic, validation and API work	Prime agentic-pattern territory: high volume and well specified
Defects and bugs (113 tickets)	Quality escapes and rework late in the flow	QA agentic widens automated coverage and regression, reducing escapes and reopened work

This follows the same prove, expand, scale shape as the current engagement. The sprint proved the model in engineering. The next phase expands it across the pod so the gains compound. Beyond that, the pattern scales across other eNVD pods, with the operating rhythm and light governance needed to make it stick.

9. Measurement, baseline and governance

This sprint is the baseline. From here the sensible move is to track a small set of measures over time, per role and per pod, so the story is measured rather than modelled at the next review.

- An adoption survey across the whole team, repeated so the trend is visible.
- Throughput measures such as work completed over a rolling period and deployment frequency, which reflect real delivery better than cycle time.
- Lightweight time-saved capture per role, so the three-hour assumption is confirmed or corrected with real data.
- Usage and cost from the built-in analytics, kept as context for governance rather than as the value story.

None of this needs heavy tooling. A self-service view that updates on its own removes the manual weekly snapshot, and anything the team wants to see differently can be produced quickly and exported to a single page when it needs to be shared upward.

10. Assumptions and next steps

Annual volume is modelled at circa 600 tickets per pod per year, with validation continuing; recent FY26 completed throughput was 483, in the same range. The main assumption still to confirm is the three-hour saving per ticket, per role, which light effort-logging on a sample of tickets during the sprint, compared against a baseline, will settle. The adoption figure depends on the survey being completed across the team, and the QA and business-analysis gains assume a named owner within your team for each function.

Immediate next steps: confirm the value assumptions with your team; run the adoption survey across the whole team; agree the small set of ongoing measures above; and scope the extension into business analysis and QA as the next phase.