# AI on Both Sides of the Pull Request

Generating from Specifications and Reviewing against Standards, with a Human in the Loop

Executive Summary

Use AI where it helps the business.

Oliver Bodemer · e-Softworks, Inc.

October 2026

Document EXE-2026-10 · Version 1.0 · October 2026

This summary derives from the scientific article *AI on Both Sides of the Pull Request* (Oliver Bodemer, e-Softworks, Inc., October 2026, document SCI-2026-10). That article presents an operating model. It does not report measured delivery speed or defect rates.

**Key Findings**

Move both duties to AI, writing the change and reviewing the pull request, or the review remains the queue. Keep a human on both sides so the two instances cannot revise each other without a bound. A finding cites one catalogue entry and one place. A remark that cites neither does not hold the pull request. The cycle stops on accept, reject, override, or the round cap. The team sets the cap. Reaching it does not merge.

## The decision

When humans both write the change and review the pull request, both of those tasks have to move to AI if the team is to keep the pace. A human still confirms intent before a pull request is opened, and a human still decides the review.

The split is the duty model this summary keeps. The writing is one duty. The review is a second duty. Moving one without the other leaves the second as the queue. Moving both keeps the two duties running at the same pace, and it keeps a human stop on each one.

The summary states the duty split, the human checkpoints, and the stop rules. A later comparison would set three setups side by side: a person writes and a person reviews; an AI instance writes and a person still reviews; both duties move to separate AI instances with a human stop on each side. Measured delivery speed and defect rates of that comparison lie outside this summary. The summary also does not price a tool. Two separate runs are enough. The cost that remains is human time on the two seats.

## The operating model

One AI instance creates the application change from specifications and requirements. A second, independent instance counter-checks the source code in the pull request against a fixed standards catalogue. The reviewer comments. It does not merge.

The generator instance reads the specification, the requirements, and the constraints. It produces one application change: the source, and the tests that demonstrate the requirements the change claims to meet. The generator's output is the change.

The reviewer instance reads two inputs: the pull request, and the standards catalogue. It writes findings. Each finding cites one entry in the catalogue and one place in the change. The reviewer instance stops at the finding. It does not edit the branch. It does not merge the pull request.

When a finding calls for a change, the generator instance makes that change on a new pass. The finding joins the specification, the requirements, and the constraints as an input to that pass. The two instances keep their hidden reasoning to themselves. What crosses the boundary is an artefact either can read: a change in one direction, a finding in the other.

## The three questions

The model answers three questions:

1. How can the generator produce a change that is complete enough to review, with a human confirming intent first?
2. How can the second instance review that pull request against the catalogue, and which findings stay a human decision?
3. Which stops keep the revision cycle bounded while both duties run at AI speed?

The first question is answered by the generator's inputs and the human checkpoint before the pull request. The generator reads the specification, the requirements, and the constraints. It produces the source and the tests. A human confirms the intent, the scope, and that the change still matches what was asked before a pull request is opened. The

seat sends the change back for invented behaviour, for scope drift, or for an unsafe shortcut. The send-back names the failure and the place.

The second question is answered by the reviewer's catalogue and the human decision on the findings. The reviewer applies five review questions: specification conformance, Clean Code maintainability, ISO/IEC 25010 reliability, security, and maintainability as review questions, named security issues, and tests tied to the claimed requirements. A human decides every material finding, and a human decides the pull request. A remark that cites no catalogue entry and no place does not hold the pull request.

The third question is answered by the four stops that bound the cycle: a human accepts the pull request, a human rejects it, a human overrides the open findings, or a round cap is reached. The team writes the count before the first send-back. The count is how many review-seat send-backs one change may have. This summary does not choose the number, because that choice belongs to the team, made before the cycle starts. If the team cannot agree the count, the cycle does not start. Reaching the cap does not merge.

## What to put in place first

A first team writes five things down before a cycle runs:

1. **The standards catalogue.** The five review questions. A finding cites one entry and one place in the change.
2. **Two separate instances.** Separate instructions, inputs, and outputs. The generator writes the change. The reviewer counter-checks it.
3. **Named human seats.** One before the pull request (intent seat), and one on the review (review seat). One person may hold both, and each seat still writes its own record.
4. **A round cap.** A count of rounds, chosen before the first send-back. Reaching the cap does not merge.
5. **The three inputs.** The specification, the requirements, and the constraints for the change at hand, written before the generator runs.

With those five things written, the cycle can run. The measurements of speed, defects, and shared misses remain outside the model.

## What this summary does not claim

This summary states the operating model. It does not report measured delivery speed. It does not report measured defect rates. It does not set the round cap number. It does not choose a model family, a repository, or how many people hold the seats.

Delivery speed, defect density, human effort, shared-miss frequency, and override rate are the measurements the three setups would need. They are not taken here. The full argument is the scientific article named above.

Oliver Bodemer · e-Softworks, Inc. · October 2026