SINGULARICS

White Paper

Ship or Learn

Every day ends in someone's hands, or with a named blocker. Never a slide. Never a status report.

Alexander S. Petty
SINGULARICS | singularics.ai

First published August 2026.

singularics.ai

Contents

Abstract

The one-day cycle ends in one of two places: in someone's hands, or with a named blocker that redirects tomorrow's spec. There is no third option. There is no slide. There is no status report. There is no recorded walkthrough offered as a substitute for the real thing. This paper defines the two destinations, the clearance test that determines which one applies, the requirements for a hands-on environment, and the learn branch that turns a blocked day into a redirected one. It also defines why establishing clearance for a class of change is among the highest-value work the Flow Council does, and why spec-and-explore cycles are legitimate planned outcomes rather than lost days. The discipline is simple and it is non-negotiable: every day produces something a human can use or it produces a named constraint that changes what happens next. Nothing sits in an ambiguous state overnight. Ship or learn. Those are the only two acceptable endings.

The Two Destinations

Ask any team lead in a large enterprise where last sprint's completed work is right now and count how long it takes them to answer. In our experience, the median response time is over thirty seconds, and the answer is usually "staging" or "I'm not sure." That uncertainty is not a knowledge gap. It is a structural failure. Work that is done but not in anyone's hands is work whose value is speculative, because nobody has used it to form a judgment.

Shipping is a destination decision. It is never a question of whether anything shipped. Every day, the increment lands somewhere. The only question is where.

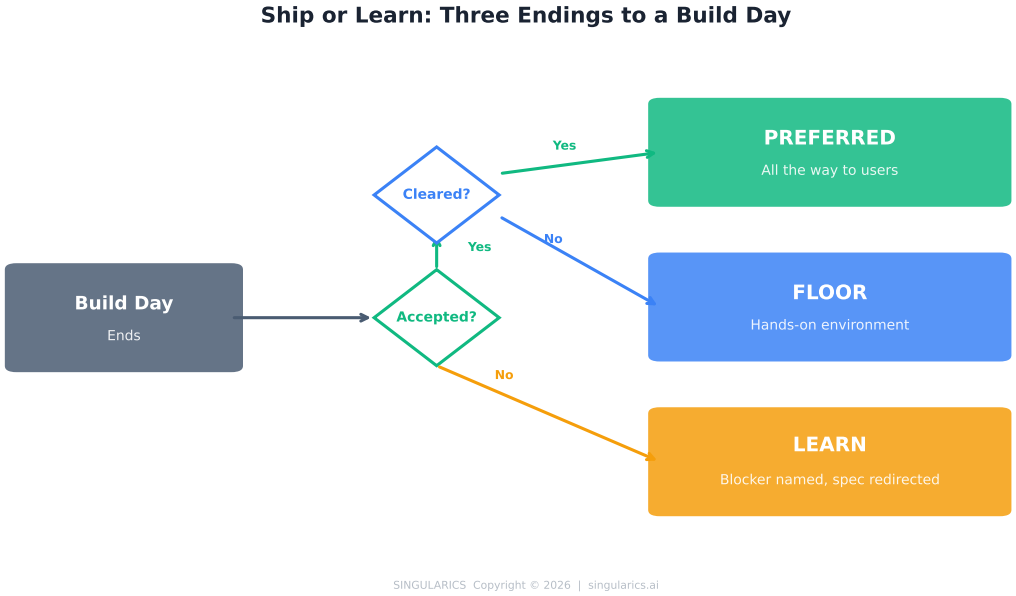

Figure 1: The three endings to a build day. Preferred: all the way to users. Floor: a hands-on environment stakeholders can reach. Learn: the increment is not accepted, the blocker is named, and tomorrow's spec is redirected.

Preferred: all the way to users

Where regulatory clearance and the delivery pipeline both allow it, the increment reaches production and real users the same day it is accepted. This is not continuous deployment for its own sake. It is the shortest path between a judgment and its consequences, which is the fastest way to learn whether the judgment was right.

An increment that sits in a staging environment for a week before reaching users is an increment whose feedback loop has been stretched from one day to eight. Every day of that stretch is a day the team might build on an assumption the users

would have corrected. The one-day cycle exists to keep that loop tight. Same-day production is what keeps it tightest.

Floor: a hands-on environment

Where same-day production is not available, the increment lands somewhere stakeholders can open it and use it themselves. This is the floor. It is not optional. It is not aspirational. It is the minimum acceptable ending to a build day.

> *Never a slide. Never a status report. Never a recorded walk-through as a substitute. The point is that a stakeholder can form an independent opinion by using the thing, which is the only reliable way to surface a misunderstanding before it becomes expensive.*

The hands-on environment is what makes daily acceptance meaningful for stakeholders who are not in the room. It converts acceptance from an internal team event into something the broader organization can verify.

Three requirements define the floor, and all three must hold simultaneously. The environment must have realistic data. It must be reachable without a scheduled session. It must be current within one build day. Miss any one and the floor becomes a demo environment: something the team uses to show work rather than something stakeholders use to form their own judgment. Section 3 defines each requirement in detail.

The decision rule is binary. At the end of the day, ask: can a stakeholder who was not in the room open a browser and use what was built? If yes, the floor is met. If no, treat it the same way you would treat a red harness: the day did not close, and the gap is named as a blocker for the Flow Council.

The Clearance Test

Whether an increment goes all the way to users or stops at the hands-on environment is determined by clearance, not by the team's confidence or the Decision Owner's preference. Clearance is established once per outcome, not per increment.

Three conditions, all of which must hold.

1 The change class has an approved deployment path. The category of change this outcome produces has been reviewed and cleared through whatever regulatory and operational gates apply. This is done once, at the start of the outcome, and covers every increment that falls within the class.

2 The control layer of the harness covers the applicable policy constraints. The automated checks that run on every push enforce the policy rules this class of change must satisfy. If the harness cannot express a constraint as a check, the constraint is unguarded and same-day production is not safe.

3 Rollback is automatic and proven. If the increment causes a problem in production, the system can return to the prior state without human intervention, and this capability has been tested, not assumed.

Where all three conditions hold, the increment ships to users. Where any one fails, the increment lands in the hands-on environment. The team does not make this call on a per-day basis. The clearance applies to the class of change, and every increment within that class follows the same path.

The Environment Bar

The hands-on environment is the floor, and it only works as a floor if it meets three requirements. Miss any one of them and it becomes a demo environment, which is something the team uses to show work rather than something stakeholders use to form their own judgment.

Realistic data

The environment must contain data that resembles what users will encounter in production. Synthetic data that covers only the happy path produces a demo experience, not a judgment experience. The data does not need to be production data, and in a regulated context it usually should not be. But it must be realistic enough that a domain expert using the environment can encounter the edge cases their business actually produces.

Reachable without a scheduled session

A stakeholder must be able to open the environment and use it without scheduling time with the team, without requesting access, and without waiting for someone to set it up. If using the environment requires the team's presence, it is a demo and not a hands-on environment.

> *The test is simple: can a stakeholder who was not in the room today open the environment tomorrow morning and use what was built? If the answer is no, the floor has not been met.*

Current within one build day

The environment must reflect the latest accepted increment, deployed within one build day of acceptance. An environment that is two weeks behind the accepted work is not a floor. It is a snapshot of a past state, and stakeholders using it will form opinions about work that has already been superseded.

The Learn Branch

Not every day ends with an accepted increment. Some days end with the increment not accepted, and that is normal. The requirement is that the failure is precise and productive.

When the increment is not accepted, three things happen before the day closes.

The blocker is named

Not "it is not quite there yet." Not "we need more time." A specific constraint is identified: which criteria failed, what evidence shows they failed, and whether the failure is in the build, the spec, or the environment.

The acceptable endings to a day where the increment is not accepted are:

- Criteria three and seven fail, and here is the evidence from the harness.
- A decision is required that the Decision Owner cannot make inside their envelope, and here is the exact question for the Flow Council.
- The spec was wrong, and here is what the demo revealed that the spec did not anticipate.

Each of these is actionable. Each points to a specific next step. "Not quite there yet" points nowhere.

Tomorrow's spec is redirected

The blocker changes what happens next. If criteria failed, tomorrow's spec addresses the failing criteria. If a decision is needed from the Flow Council, tomorrow's spec is held until the decision arrives. If the spec was wrong, tomorrow is a spec correction day.

This is the learn in ship-or-learn. The day did not produce a landed increment, but it produced a named constraint that changes the plan. That is learning, and it is a legitimate, planned outcome rather than a lost day.

The spec-and-explore cycle

When a Replace happens at daily planning and the new direction is not yet clear, do not force a build day. Run tomorrow as a spec-and-explore cycle. The fleet investigates, prototypes throwaway options, and gathers what the Decision Owner needs to decide. The day still ends with something in someone's hands, but the artifact is a decision rather than an increment.

> *A spec-and-explore day is a planned outcome, not a lost day. The artifact is a decision, not an increment. That decision redirects everything that follows, which is more valuable than an increment built on an assumption the demo just invalidated.*

The cycle works the same way as a build day. It starts with a spec, but the spec's acceptance criteria describe what the team needs to learn rather than what it needs to build. The harness still runs, but it validates the exploration rather than the delivery. The Decision Owner still accepts, but what they accept is a decision about direction rather than a landed increment.

This is the learn branch working at full fidelity. The daily cycle does not break when the plan is wrong. It absorbs the surprise, converts it to a named constraint, and redirects.

Nothing sits in an ambiguous state overnight

At the end of every day, the status of every item in the Now horizon is one of three things: accepted and landed, not accepted with a named blocker, or in flight with a spec signed for tomorrow. There is no fourth state. There is no "in progress" that carries over without a decision.

> *A day that ends with "we will look at it tomorrow" has not ended. It has been abandoned. The discipline of naming the blocker is what converts an inconclusive day into a productive one.*

This is the non-negotiable discipline at the heart of ship-or-learn. Ship means the increment reached users or a hands-on environment. Learn means the blocker was named and tomorrow's spec was redirected. Both are legitimate endings. Everything else is drift, and drift compounds overnight into confusion that consumes tomorrow's spec time, the Decision Owner's availability, and the Fleet Lead's judgment. The two-destination rule prevents that compounding by forcing a clean close every day.

The counterargument is that some work genuinely cannot be evaluated in a single day. That is true, and the model handles it explicitly: if the outcome requires four months, it turns the cycle 80 times rather than running one long build. The constraint is not that the outcome must be small. It is that each day's increment

must be judgeable. A four-month outcome built in daily increments is more controllable than a four-week outcome built in one batch, because the Decision Owner has 80 opportunities to correct course rather than one.

Two edge cases that test the binary

The multi-day increment, a data migration or an infrastructure cutover that cannot reach users in a day, is sliced by provable states instead of features. The day the rehearsal runs clean in the hands-on environment is a day that landed, because the evidence is in someone's hands even though the cutover is not. And the increment nobody can see, the backend contract with no screen, lands as something technical hands can use. A working endpoint a stakeholder's analyst can call, a dataset diff a domain expert can read. In someone's hands includes those hands.

The Never-a-Slide Rule

The prohibition on slides and status reports as substitutes for hands-on artifacts is not a preference. It is a structural requirement of the model.

A slide describes work. A hands-on environment contains work. The difference is that a description is filtered through the presenter's understanding, while the artifact is available for independent examination. When a stakeholder uses the actual thing, they discover what the presenter did not mention, what the presenter did not notice, and what the presenter assumed was obvious. These discoveries are the highest-value feedback the organization can receive, and they are invisible in any mediated format.

100% of days ending in something a human can use. This is the outcome indicator that measures whether the model is working. Not velocity, not story points, not agent output volume. Percentage of days ending in hands.

Status reports create a second problem. They consume the Decision Owner's time, which is the scarcest resource in the model. A Decision Owner who spends an hour preparing a status deck has lost an hour of spec work or availability. The hands-on environment is the status report. Anyone who wants to know the status can use the environment.

Establishing Clearance

Establishing clearance for a class of change is among the highest-value work the Flow Council does, because it converts every future increment in that class from a queued release into a same-day one. Every increment that ships same-day instead of waiting in a release queue is an increment whose feedback loop stays at one day. Every increment that waits is an increment whose feedback loop stretches to however long the queue is, and the team builds on unvalidated assumptions for the duration.

This is why clearance work belongs to the Flow Council and not to a release management function. The Flow Council holds the portfolio view and can see which classes of change produce the most increments. Clearing the highest-volume class first produces the largest return. That sequencing decision is a portfolio decision.

One clearance, worked

An internal servicing dashboard was classified as a change class in week one. Reversible, no external commitments, policy checks in the control layer, rollback rehearsed once in front of the council. Every increment on that Effort shipped same-day for the rest of the outcome, and nobody discussed deployment again. The payout calculation path on the same portfolio was deliberately not cleared. Those increments landed in the hands-on environment while their harness evidence accumulated into the case the compliance function needed for the next clearance conversation. Same model, two speeds, both of them chosen rather than suffered.

The clearance bar

The Flow Council works with the regulatory and operational governance functions to identify which classes of change can be cleared. For each class, the three conditions from the clearance test in Section 2 are verified: approved deployment path, harness coverage of policy constraints, and proven rollback.

This work is done once per class, not once per increment. The investment is front-loaded, and the return is every increment thereafter shipping the same day rather than waiting in a release queue.

Concretely, the Flow Council does four things for each candidate class. First, it identifies the regulatory and operational gates that apply to changes in that class. Second, it works with the relevant governance functions to establish an approved deployment path. Third, it verifies that the harness's control layer covers the

applicable policy constraints for that class. Fourth, it proves rollback: the system returns to the prior state automatically, and this has been tested, not assumed.

The compounding

Each class of change that is cleared reduces the number of increments that stop at the hands-on environment. Over time, the proportion of increments that reach users on the day they are accepted grows. This is the metric that measures pipeline and clearance maturity rather than team performance.

> *Getting one class of change cleared for same-day production changes the conversation more than any presentation. It is proof that the model works, delivered in the form that the model demands: something in someone's hands.*

The compounding is arithmetic. If a portfolio produces twenty increments per week and five classes of change cover 80% of them, clearing those five classes converts sixteen increments per week from queued releases to same-day production. Each conversion tightens one feedback loop from days or weeks to one day. The aggregate effect on learning rate is substantial, and it accumulates permanently.

16 increments per week shifted from queued releases to same-day production when five change classes are cleared in a twenty-increment portfolio. Each cleared class is permanent. The investment is front-loaded; the return is every increment thereafter.

Behind the gate

Some classes of change will never be cleared for same-day production, and the model does not pretend otherwise. Changes that require regulatory interpretation, risk appetite changes, or new external commitments follow their existing governance path. The model does not compress these, and claiming otherwise is how transformations lose credibility with second-line oversight.

The discipline is to be honest about which classes are clearable and which are not, rather than treating all work as if it falls into the same category. The harness cannot express every constraint as a check, and where it cannot, the constraint

is unguarded. An honest assessment of what the harness can and cannot cover is the foundation of clearance. Overstating coverage to get clearance is how you produce an audit finding rather than a same-day deployment.

The Release Train, the Holding Area, and the Status Report

Ship-or-learn replaces three things that exist in most organizations today.

The release train. A cadence-based release schedule batches increments and ships them together, typically every two weeks or monthly. The batch exists because the cost of releasing is high enough to justify amortizing it. When the clearance test passes for a class of change, the cost of releasing drops to near zero, and the batch no longer has an economic justification. Each increment ships on its own day rather than waiting for a train.

The staging environment as a holding area. In most organizations, staging is where work waits between acceptance and production. That wait stretches the feedback loop and creates a gap between what was accepted and what users experience. The hands-on environment in this model is not staging. It is the floor: the minimum acceptable ending to a day, not a waypoint on the path to production.

The status report. When the work is in someone's hands, the status is visible by using it. The status report exists to describe work that cannot be seen, and when the floor requirement is met, the description is redundant. This is not an argument for less communication. It is an argument for communication through the artifact rather than through a document about the artifact.

Failure Modes

Three failure modes from the operating model's failure catalog apply directly to ship-or-learn.

Demo-only acceptance. Tell: acceptance happens because it looked right in the demo. A demo shows one path through the system. The harness checks every path, every time. Acceptance based on a demo alone is acceptance based on incomplete evidence. Correction: return to evidence. The acceptance decision is made against the harness results and the spec criteria, not against a walkthrough.

Harness not in CI. Tell: the harness runs before demos rather than on every push. A harness that runs sometimes is a suggestion. A harness that runs on every push

is a gate. The difference is enforcement, because a suggestion can be overridden when the team is under pressure, and pressure is exactly when the gate is most needed. Correction: move it into the pipeline. Until it runs on every push, the clearance test cannot pass.

Slices too big. Tell: the Replace rate at daily planning exceeds 30%, or increments regularly span more than one day. When slices are too big, the learn branch fires constantly, and the team spends more time replanning than building. Correction: halve the slice. Re-run the checkability test on every criterion. If the spec does not fit on a page, it is two specs. A sustained Replace rate above 30% is the diagnostic that the slices remain too large or the work is still exploratory.

How Ship or Learn Closes the Loop

Ship-or-learn sits at the end of the daily cycle and depends on two operating mechanisms.

The harness. The clearance test requires that the control layer of the harness covers applicable policy constraints. The harness is what makes same-day production safe in a regulated context, and it is the Decision Owner's instrument for accepting without reading code.

The one-day cycle. Ship-or-learn is its closing step. The the one-day cycle. The cycle defines what happens before this point: spec, build, judge. The closing decision is what happens at the end: land or redirect. Together they form the complete daily loop.