Constraint Snapshot

Facilitator guide

Your job: map one slow piece of work, mark and size the waits between the steps, and leave with the biggest wait circled as the stall and one owned move aimed at it. Keep it fast and rough — the map is a means, not the point.

The one idea: work doesn't slow down where people are working. It slows down where work is waiting — in the handoffs and queues between the busy steps, where nobody is watching. This activity makes those waits visible, so the team fixes the real delay instead of working harder on the parts that already move.

· WHEN DO I USE THIS?

Use this when

GOOD FIT

Delivery feels slow and you can't say why. Everyone's busy, but the work takes far longer end to end than the hands-on time would suggest — and no single cause stands out. You can name one real piece of work that moved slowly, start to finish.

WRONG TOOL

Work has stalled and you need the blocker now › Dependency Snapshot. Nothing's stuck yet, you're defending a deadline › Protect the Critical Path. The load is simply too big › Capacity Triage. Too much work is entering to begin with › Execution Queue Control.

The one test before you start: can you pick one real piece of work that moved slowly, end to end? You map where its time went — not the whole system. If you can't name one real item, there's nothing to map.

· WHAT WILL THE TEAM LEAVE WITH?

What you'll leave with

WHAT YOU GET

A simple map of how the work flows, the waits marked and sized, and the worst stall circled — with one owned move aimed at shortening it. The one delay that, removed, shortens overall delivery the most.

WHY IT MATTERS

The team sees that most of the delay was waiting, not working — and that the worst stall was sitting in a gap nobody owned. Delivery stops feeling mysteriously slow, and the real constraint becomes something you can point at and fix.

© Witmer

PART 1 BEFORE THE ACTIVITY

get ready

1 · WHAT IS MY JOB HERE?

Your job

You are running this to one outcome: the biggest wait in the work circled as the stall, with one owned move aimed at it. Your job is to:

- get the team to map the **waiting, not the working** — the delay is in the gaps between busy steps, not in the steps themselves
- keep the map **fast and rough** — this is a stall map, not a process diagram
- make them **size every wait** — without sizes there is no biggest wait, and no stall
- hold them to **one owned move** at the biggest wait — not a list of five, and not a fix aimed at a busy step

THE ONE RULE

Mark the waiting, not the working. A wait is time when nobody is working the item — between steps, in a queue, awaiting a handoff or sign-off. If you map the busy steps you find nothing; if you map the waits you find the stall. The biggest wait, not the busiest step, is the constraint.

2 · WHO DOES WHAT?

Roles

Role	What they do
You (facilitator)	Run the activity and keep the team on the waiting, not the working. Keep the map fast and rough, run the wait tests on each gap, keep time, stay neutral, and protect the one move at the end.
Participants (3–6)	The people who actually moved the work. They lay out the steps, mark and size the waits, name the worst stall, and agree the move. 25–30 minutes.

3 · HOW DO I OPEN?

Opening script — 2 minutes

In your own words, never read from the app page. It sets the frame without giving away the reveal.

SAY — IN YOUR OWN WORDS

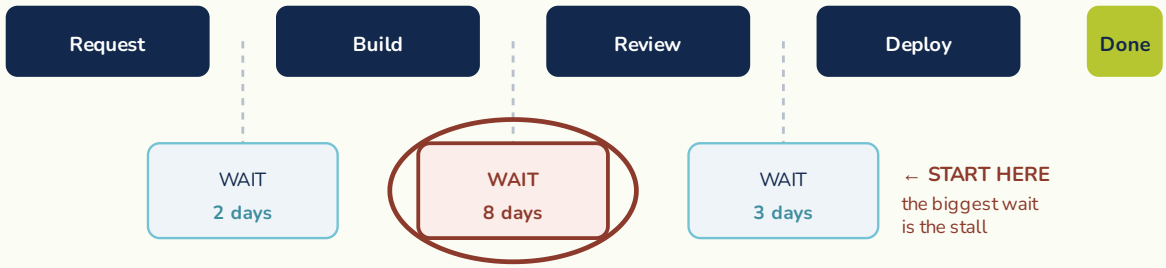
“We’re going to take one real piece of work that took too long, and map how it actually moved — step by step, request to done. But we’re not mapping the busy parts. We’re marking every place the work just *sat* — waiting for a review, a sign-off, a handoff. Then we’ll size those waits and see which one is really costing us. My bet is it won’t be the step everyone complains about.”

© Witmer

4 · HOW DO I PREPARE THE SPACE?

Setup and materials

THE BOARD AT A GLANCE — STEPS ACROSS, WAITS DROPPING BELOW



WORKING TIME	Build 2 days · Review 4 hrs · Deploy 2 hrs — a few days of actual work

Mark the waiting between steps, not the busy steps themselves.

Days of waiting sit around hours of working — and the biggest wait is where delivery is really lost.

What	Why it's there
One piece of slow work	A real item that took too long, request to done. Steps 1–3: this is what you map. If you can't name one, there's nothing to map.
A whiteboard or wall	Room to lay the steps left to right, with the waits dropping below. Step 1: the steps go up here as a row.
Sticky notes + marker	One note per step and one per wait; the marker sizes each wait and circles the stall. Steps 2–5: this is what leaves the room in the photo.

A note on the term: a **stall** is the biggest wait in the flow — the single place the work sat longest with no one touching it. It is the constraint. Everything in this activity is aimed at finding it and shortening it.

PART 2 DURING THE ACTIVITY

run it

Before you begin — 30-second check

- ✓ There's one real, slow piece of work to map — request to done
- ✓ The people who actually moved the work are in the room
- ✓ The wall is clear to lay steps left to right, with room for waits below
- ✓ You're ready to hold the rule — mark the waiting, not the working

5 · HOW DO I RUN IT?

How to run it

Brief the team (2–3 min), then run these in order. The bold part is what you say or do; the note under it is where you earn your keep.

© Witmer

1. Pick one piece of work	**3–4 min**	*"What's one real thing that took too long, request to done?"* Choose something real that moved through the team start to finish. If they can't name one, there's nothing to map.
2. Map how it moved	**5–6 min**	*"Walk the work from request to done — what are the steps?"* Put each on a note, left to right. Map what actually happened, not what the process document says should happen. Keep it fast and rough; 5–8 steps is plenty.
3. Mark the waiting	**4–5 min**	*"After this step finished, how long before the next one started?"* At each point where the work sat and waited — between steps, in a queue, awaiting a handoff — mark it below the gap. Mark every wait, even a few hours; small waits add up. If you're unsure whether something is a wait, run the Hands test (next page) at once rather than debating it. This is where the tool lives.
4. Size the waits	**5–6 min**	*"How long did it sit here — hours, days?"* Estimate each wait beside its mark. Rough estimates are enough; you're after proportion, not precision. Don't argue about exact numbers — relative size is what matters.
5. Name the worst stall	**3–4 min**	*"Which wait is the biggest?"* The biggest wait is the constraint — not the busiest step. Circle it. It's usually a queue or handoff, not the step everyone complains about. Let the contrast — days of waiting around hours of working — land.
6. Decide one move	**5 min**	*"What one change would shrink that wait?"* Choose the smallest change most likely to reduce the biggest wait, and who owns it. Give it a rough when. Photograph the board before anyone leaves.

> **These times are indicative** and will depend on the nature of the problem and the size of the team. Step 2 (mark and size the waits) is where the value is — protect it. Step 4 (one move at the biggest wait) is what gets skipped; protect it too. A stall with no move is just a diagram.

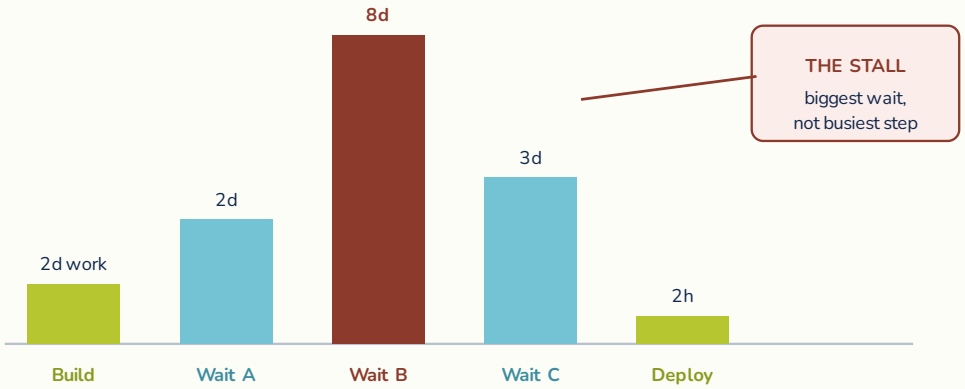
6 · HOW DO I TELL A WAIT FROM WORKING TIME?

The wait tests

Don't rely on judgement — apply these at each gap. Teams instinctively mark the busy steps; the delay is in the waits between them.

© Witmer

SIZE EVERY WAIT — THEN THE BIGGEST ONE IS OBVIOUS



Test	Ask the team	Verdict
1 · Hands	“Was anyone actively working it, or was it just sitting?”	Just sitting › it’s a wait
2 · Calendar	“Did days pass with no one touching it?”	Yes › it’s a wait
3 · Blame	“Is this the step everyone blames?”	Usually working, not the wait

The stall rule: mark only the gaps where work sat with no hands on it. Then the biggest wait — not the busiest step — is the stall. Don’t argue about exact durations; relative size is enough. If the team’s instinct points at a busy step, run the Hands test: busy is not the same as slow.

When to step in — and when to stay out · a principle, not a step

STEP IN

They mark the busy steps, not the gaps → redirect to the waits.

They start a detailed process map → keep it fast and rough.

The move aims at a busy step, not the biggest wait → point it at the wait.

They skip sizing the waits → without sizes there’s no stall.

STAY OUT

The map looks messy → fine, if the waits are marked.

Someone’s surprised the busy step wasn’t the problem → let it land.

They debate exact durations → roughly right is fine.

The map is quick → keep it fast and rough, don’t pad it.

© Witmer

The clearest tell it's time to step in: the team is busy improving the step it always blames, and no one has marked a single wait. Stop them: "where did the work sit — with no one touching it?" The stall is in the gaps.

PART 3 AFTER THE ACTIVITY

make it stick

7 · HOW DO I CLOSE?

Closing script — 2 minutes

Reconstruct the discovery — don't summarise it. Replay where the team started so the insight lands as something they lived.

SAY — IN YOUR OWN WORDS

"When we started, most of us would have bet the slow part was *[the busy step]*. But look at the board: that step took hours. The work then sat *[X days]* waiting in a queue nobody owned. That's the stall — and that's the one wait we're going to shorten. Most of our delay was waiting, not working."

8 · DID WE GET A GOOD RESULT?

What good looks like

A GOOD RUN

A rough map with the waits marked and sized, the biggest wait circled, and one owned move aimed at it. The team can say: "most of our delay was waiting, not working."

A HOLLOW RUN

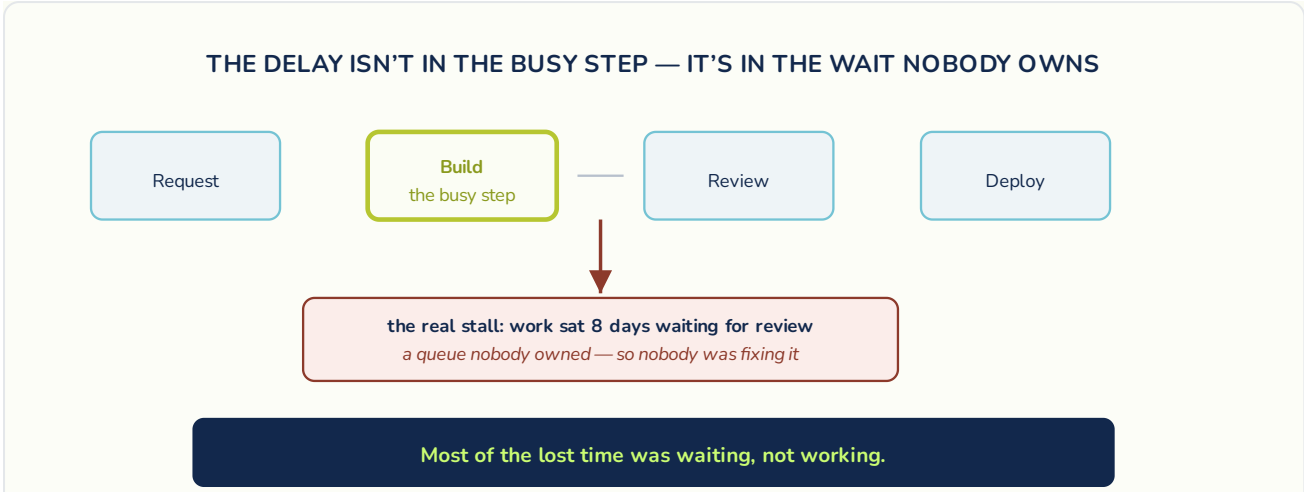
A detailed process map with no waits marked; the team "fixes" the busy step it always blames. If so, you mapped the working instead of the waiting. Go back and ask: "where did the work sit, with no one touching it?"

9 · WHY DOES THIS WORK?

The idea underneath

People naturally focus on the work they can see, because that's where the effort and the complaints are. But work usually spends far more time waiting between steps than being worked on — in handoffs, approvals and queues. Those waits are invisible, so they never get fixed. Marking the waiting separately from the working makes the real delay show up — and that's usually where the biggest improvement comes from.

© Witmer



Before you finish — the check

- ✓ The steps are laid out, request to done — what actually happened, not the documented process
- ✓ A wait is marked after every step where the work sat
- ✓ Every wait is sized — hours or days
- ✓ The biggest wait is circled as the stall
- ✓ One owned move is aimed at the stall, with an owner and a rough when
- ✓ The board has been photographed before anyone leaves

10 · HOW DO I MAKE THIS A HABIT?

Take it back to work

THE HABIT

When delivery feels slow: "Where is the work waiting — not where are people busy?"

The lasting change: the team stops working harder on the busy steps and starts asking where the work is waiting. Run it after a slow release, a customer escalation, or a quarterly review — it becomes a habit, not a one-off. Speed comes from reducing waiting, not increasing effort.

© Witmer

Appendix · Reference

You should not need this while you run the activity. Read it before, or use it if a run goes sideways.

11 · A WORKED RUN

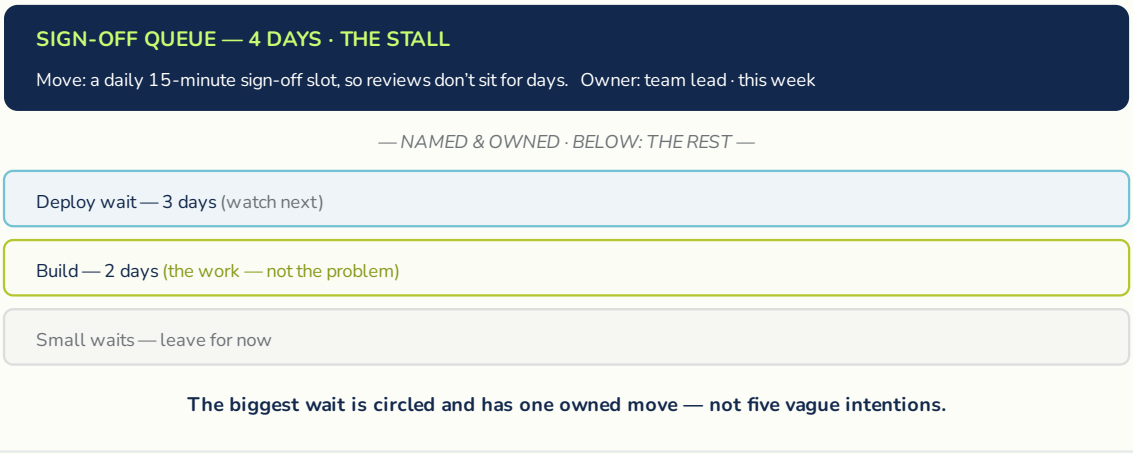
Watch the reveal land

A team mapped a single customer feature, request to shipped. Everyone “knew” the slow part was the build.

THE MOMENT THAT FOUND THE REAL STALL

Team: “The build is the bottleneck — it’s where all the work is.”
Facilitator: “How long was the build actually being worked on?”
Team: “…about two days of hands-on work.”
Facilitator: “And after it was built — how long before it shipped?”
Team: “…it sat four days waiting for sign-off, then three more waiting to deploy.”

FINISHED STALL MAP — ONE CIRCLED, ONE OWNED MOVE



The biggest wait is circled and has one owned move — not five vague intentions.

What the facilitator does: circles the four-day sign-off wait. “So two days of work, seven of waiting. The stall is the sign-off queue — not the build.” They didn’t argue about the build; they sized the waits and let the timings end the debate. The team had been trying to speed up the wrong thing.

© Witmer

12 · WHAT GOES WRONG MOST OFTEN?

Common mistakes

Mistake	Why it fails the run
Mapping the busy steps	The commonest failure. The delay is in the gaps between steps, not the steps. Mark only where work sat with no hands on it.
Skipping the sizes	Without a rough size on each wait there is no biggest wait — and no stall. Hours or days is enough.
A detailed process diagram	This is a stall map, not process mapping. Keep it fast and rough; 5–8 steps.
A move aimed at the busy step	If the move speeds up a busy step rather than shrinking the circled wait, it's aimed at the wrong place. Point it at the wait.
Five moves, none owned	End with one owned move at the biggest wait — not a list of intentions. Fix one constraint, then measure again.

13 · WHAT IF SOMETHING GOES WRONG?

Edge cases

What happens	What to do
Every step looks busy, no waits found	Look harder at the handoffs — between people, teams, or tools. The waits hide in the gaps no single person owns.
The biggest wait is outside the team	Common and useful. The move becomes "escalate or renegotiate the queue with whoever owns it," with an owner on your side.
Two waits look equally big	Size them properly first; if still tied, pick the one the team can actually act on, and note the other to watch.
The team insists the busy step is the problem	Run the Hands test on it. If people were genuinely working it the whole time, fine — but compare its hours to the waits' days before concluding.
They want to map the whole system	Don't. One piece of slow work, one pass. Mapping everything kills the speed and buries the stall.
No single wait dominates	Sometimes the delay is many small waits, not one stall. Name that pattern: ask whether the real constraint is the accumulation of many small handoffs. Then choose one handoff to improve first — the fix is handoff discipline, not fixing all of them at once.

© Witmer

14 · WHERE DOES THIS FIT?

Where this sits in the Gym

If the question is...	Use this
“What are we waiting for?” — the work is already stuck	**Dependency Snapshot** — trace the stuck work to its source blocker.
“What must not be interrupted?” — before the cycle	**Protect the Critical Path** — protect the pacing work before it’s at risk.
“The load is simply too big” — too much in play	**Capacity Triage** — decide what to shed when there’s more work than capacity.
“Too much work is entering to begin with”	**Execution Queue Control** — manage what gets in, before it ever stalls.

Rule of thumb: Constraint Snapshot finds where time leaks inside one flow; Dependency Snapshot finds who you’re waiting on when work has stopped. Slow versus stuck.

© Witmer