Protect the Critical Path

Facilitator guide

Your job: get the team to name the few things that pace the deadline, protect them — work, people, time — and leave with one rule for what yields when something new competes. Keep identification fast, protect the people and time (not just the task), and force a real yield rule. You are not re-planning the delivery; you are protecting what sets its date.

The one idea: teams don't miss deadlines because they can't spot the critical work — they miss them because they keep sacrificing it to whatever seems most urgent. Urgency is loud; pacing work is quiet. Decide what yields before the pressure arrives, and the choice is already made when it does.

WHEN DO I USE THIS?

Use this when

GOOD FIT

A delivery with a real, dated deadline, and the same people pulled between competing demands. The work that sets the finish date keeps getting bumped by "urgent" asks. Run it **before** the cycle, to decide what must not be interrupted.

WRONG TOOL

No firm date yet › pin the date first, there's nothing to protect. Work has already stalled › Dependency Snapshot. The whole list is too big and must be cut › Capacity & Feasibility System — this protects a path, it doesn't shrink a backlog.

The one test before you start: is anything stuck yet, or are you protecting it from getting stuck? This is proactive — run it before the cycle, to decide what must not be interrupted. If work has already stalled, you want Dependency Snapshot instead.

WHAT WILL THE TEAM LEAVE WITH?

What you'll leave with

WHAT YOU GET

A protection charter: the pacing work in a protected column — with the people and time it needs — the rest in a yields column, and one yield rule for when new work competes.

WHY IT MATTERS

The deadline-setting work stops getting interrupted. "Urgent" asks have a clear answer — they wait. The team stops re-prioritising every week, because the rule is already decided.

© Witmer

PART 1 BEFORE THE ACTIVITY

get ready

1 · WHAT IS MY JOB HERE?

Your job

You are running this to one outcome: a protection charter the team will hold. Your job is to:

- keep identification **fast** — run the pacing tests, don't debate importance
- protect a **few** things, not many — if everything's protected, nothing is
- protect the **people and time**, not just the task
- force a **real yield rule** — a charter with no rule is just a diagram

THE ONE IDEA

Decide what yields before the pressure arrives. Urgency is loud; pacing work is quiet. Left to the week's noise, teams protect whatever seems most urgent — and the work that actually sets the date gets interrupted. Deciding in advance what is protected and what yields means the choice is already made when the pressure arrives, so the critical path holds up when the urgent work arrives.

2 · WHO DOES WHAT?

Roles

Role	What they do
You (facilitator)	Run the flow, run the pacing tests, hold the yield rule, and keep time. You protect the process; you don't decide the delivery's content.
Participants (a team)	The people doing the delivery. They find the pacing work, sort protected versus yields, and agree the rule. 30–45 minutes.
The line-holder	Named on the charter — usually the delivery owner. The person authorised to defend the charter when competing work arrives; who an "urgent" ask has to get past.

3 · HOW DO I OPEN?

Opening script — 2–3 minutes

In your own words, never read aloud. Don't assume anyone read the app page.

SAY — IN YOUR OWN WORDS

"We've got a real deadline, and every week something more urgent-feeling pulls us off the work that actually sets it. Today we're going to name the few things that pace this delivery, protect them — the work, the people, and the time — and put everything else in a column that yields. Then we write one rule for the next 'urgent' ask, so we decide now, not under pressure. The point isn't a diagram; it's a rule we'll actually hold."

© Witmer

4 · HOW DO I PREPARE THE SPACE?

Setup and materials

THE PROTECTION CHARTER — TWO COLUMNS AND ONE RULE

the deadline that paces everything, written across the top

PROTECTED — cannot be interrupted

· the pacing work (sets the date)

· the people on it — not reassigned

· the time it needs — protected

a few things, with people and time — not a list

YIELDS — gives way

· every "urgent" ask not on the path

· meetings the path doesn't need

· nice-to-haves — deferred, not killed

important work lands here too — that's the tool working

THE YIELD RULE

New work yields to Protected unless it moves the deadline itself. Name who holds the line.

Component	What it is, and when you use it
A real, dated delivery	A deadline that matters, with competing demands on the same people. If nothing is competing, there's nothing to protect.
A board or wall	Two columns — Protected and Yields — side by side, with the deadline written across the top.
Sticky notes + marker	One note per piece of work, so items move between columns. The marker writes the yield rule where the team will see it.

PART 2 DURING THE ACTIVITY

run it

Before you begin — 30-second check

- ✓ There's one real, dated delivery to protect — not a hypothetical
- ✓ The people doing the delivery are in the room
- ✓ The board's ready: two columns, deadline across the top
- ✓ You're ready to run the pacing tests out loud — and force a concrete yield rule

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 add the most.

© Witmer

1. Name the deadline that matters	**4–5 min**	*"What's the one delivery we're protecting, and its date?"* Write it at the top where everyone sees it.
2. Find what paces it	**7–9 min**	*"Which work actually sets this date?"* For each candidate the team names, run the three pacing tests below — out loud, as a team. A candidate has to survive all three to stay on the path. Keep it to a few; most work isn't pacing.
3. Build the protected column	**7–9 min**	*Write three lines, not one.* For each surviving item: the task, the people (by name), and the time (as a specific commitment). Ask the operational question: "What meetings or recurring commitments must these people come out of for this protection to be real?" A protected task with no protected people or time is not protected.
4. Build the yields column	**7–9 min**	*"What does not move the deadline?"* Everything else — requests, meetings, nice-to-haves — goes in Yields. Sort by "does this move the deadline?", not "is this important?" Yielded work is deferred, not killed — say so.
5. Write the yield rule	**5–6 min**	*"New work yields to Protected unless it moves the deadline itself."* Write it on the charter and name who holds the line. 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; the whole activity runs 30–45 minutes. Step 2 is where the tool lives — without the tests, one facilitator protects three things and another protects twelve. Step 5 (the yield rule) is the one that gets squeezed; protect it. A charter with no rule for next week's "urgent" ask is just a diagram.

6 · HOW DO I DECIDE WHAT PACES THE DEADLINE?

The pacing tests

Don't rely on judgement or on which work feels important — run these on each candidate, out loud. A candidate stays on the protected path only if it survives all three.

© Witmer

THE PACING TESTS — RUN ON EACH CANDIDATE, OUT LOUD

1 · Slip	“If this slipped a week, would the delivery date move?”	**No › not the path**
2 · Finish	“If it were done tomorrow, could we still miss the date?”	**Yes › not the path**
3 · Absorb	“Could another activity absorb its delay?”	**Yes › not the path**

KEEP only if it survives all three: slipping it moves the date, finishing early doesn’t save it, and nothing else can absorb its delay. Most work fails at least one.

The keep rule: a candidate stays on the protected path only if it survives all three — slipping it moves the date, finishing it early doesn’t save the date, and nothing else can absorb its delay. Most work fails at least one. More than about five survivors? Run the tests again, harder — the column should be short.

If two protected items always move together because one depends entirely on the other, treat them as one protected path, not two separate items. It keeps the column short and stops you double-counting the same piece of work.

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

STEP IN

Protected keeps growing → push back to a few.

They protect the task but not the people or time → an unprotected person is an unprotected path.

The yield rule is vague → force it concrete.

They confuse loud with pacing → run the slip test.

STAY OUT

Identifying the pacing work is quick → quick is the point, don’t pad it.

Putting real work in Yields feels uncomfortable → that discomfort is the tool working.

They debate exact items → roughly right is fine.

Someone resists parking their own work → let the rule decide.

The clearest tell it’s time to step in: Protected has eight things in it. That’s not protection — it’s a to-do list. Push: “if you could only protect three, which set the date?”

© Witmer

PART 3 AFTER THE ACTIVITY

make it stick

7 · HOW DO I CLOSE?

Closing script — 2 minutes

Point the team at the rule, not the diagram — the rule is what they carry out of the room.

> SAY — IN YOUR OWN WORDS
>
> "We started with a pile that all felt critical. The tests cut it to *[the few]* that actually set the date — those are protected, with the people and time they need. Everything else yields: deferred, not killed. And here's the rule — new work yields unless it moves the deadline itself, and *[name]* holds the line. That's what lets us say no to next week's 'urgent' ask without re-arguing it."

8 · DID WE GET A GOOD RESULT?

What good looks like

FINISHED PROTECTION CHARTER

The deadline that paces everything: launch ships in 3 weeks

PROTECTED — cannot be interrupted

- Payments integration (sets the date)
- Priya & Dev — full allocation, not reassigned
- Mornings blocked out, out of standing meetings

task + people + time — all three

YIELDS — gives way

- The dashboard request (doesn't move the date)
- "Quick" favours pulling the engineers off
- Meetings the path doesn't need

deferred, not killed

THE YIELD RULE

New work yields to Protected unless it moves the deadline itself.
Line-holder: the delivery owner. Charter kept visible where requests arrive.

A GOOD RUN

The protected column is short — a few items, each with people and time. The yield rule is concrete enough to say no to a real request next Tuesday without re-opening it. The team can say: "we'd have protected all eight and slipped anyway."

A HOLLOW RUN

Eight things "protected", or the task protected but not the people, or a vague rule ("we'll be disciplined"). If so, the charter won't survive the first "urgent" ask — run the tests harder and force the rule concrete.

9 · WHY DOES THIS WORK?

The idea underneath

Urgency is loud; pacing work is quiet. Left to the week's noise, teams protect whatever seems most urgent — and the work that actually sets the date gets interrupted. Deciding in advance what is protected and what yields means the choice is already made when the pressure arrives, so the critical path holds up when the urgent work arrives.

© Witmer

THE CORE INSIGHT

The output isn't the diagram; it's an operating rule the team runs on for the rest of the cycle. You're deciding *once* — in advance — instead of re-deciding every time something urgent appears.



Watch for "everything affects the deadline": it sounds reasonable and it quietly defeats the yield rule. Hold the tighter test — does this change the committed delivery date, directly? If the link is vague, it yields.

Before you finish — the check

- ✓ The protected column is short — a few items that survived the pacing tests
- ✓ Each protected item has its people (by name) and time (a concrete commitment)
- ✓ Real, valued work is sitting in Yields — and the team knows it's deferred, not killed
- ✓ The yield rule is written on the charter, concrete enough to use next Tuesday
- ✓ The line-holder is named — the person an "urgent" ask has to get past
- ✓ The charter has been photographed and will stay visible where requests arrive

10 · HOW DO I MAKE THIS A HABIT?

Take it back to work

THE HABIT

When new work arrives: "Does this move the deadline? If not, it yields to the protected path."

The lasting change: the team stops protecting whatever feels urgent and starts protecting what sets the date — because the rule was decided before the pressure arrived. Decide what yields before the pressure arrives.

© 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 facilitator move

Three weeks to ship a launch; the payments integration paced it, but “quick” favours and a new dashboard request kept pulling the two engineers off it. The moment that decided the charter:

THE RUN

Team: “These eight all feel critical — we should protect all of them.”
Facilitator: “Run the slip test on each. If it slipped a week, would the launch date move?”
Team: “…only payments, the API contract, and QA sign-off. The other five wouldn’t move the date.”
Facilitator: (moves the other five to Yields) “Then those three are the path. The rest yield — deferred, not killed.”
Team: “But marketing keeps asking for the dashboard…”
Facilitator: “Does the dashboard move the launch date? No? Then it yields too — and we write the rule so you don’t re-argue it next week.”

What the facilitator did: they didn’t argue about importance — they ran the test and let it decide. Eight became three the same way it would for any facilitator who runs the test. That exchange *is* the activity. Then one reflection question to close the loop: “If we’d had this charter from day one, which interruptions would never have happened?” That connects the exercise back to daily work.

12 · WHAT GOES WRONG MOST OFTEN?

Common mistakes

Mistake	Why it fails the run
Protecting eight things	That’s a to-do list, not protection. Push back to the few that set the date.
Protecting the task but not the people or time	An unprotected person is an unprotected path. Write all three lines.
Sorting by “is it important?”	The test is “does it move the deadline?” Important work yields too.
Leaving with a diagram and no yield rule	The rule is the output; the diagram is just the working.

© Witmer

13 · WHAT IF SOMETHING GOES WRONG?

Edge cases

What happens	What to do
A senior stakeholder demands an exception (“this is urgent — do it”)	Apply the same test, out loud: does it directly move the committed delivery date? Yes › re-charter openly. No › it yields, like anything else. Escalate to the named line-holder if the pressure persists. This is the most common way a charter dies; the rule has to survive power, not just process.
The team is genuinely split — half say it paces the date, half don’t	Don’t vote. Run the three pacing tests on it together; they usually settle it. If still split, the deadline-owner decides and the charter records the call — then move on. A decided-enough charter now beats a perfect one that never finishes.
Wants to protect everything	“If you could only protect three, which set the date?” That’s the column.
A real emergency arrives later	The rule handles it: does it move the deadline? Yes › re-charter. No › it yields.
The committed delivery date changes mid-cycle	Rebuild the charter, don’t patch it. If the date has moved, the pacing work may have changed too — re-run the tests rather than editing the old columns. Planned re-planning deserves a fresh charter.
Protected people pulled off anyway	The real test — usually a permission problem. The line-holder needs backing to say no upward. Name that.
“This is just prioritisation”	No — prioritisation ranks everything; this protects a few and lets the rest flex, with a rule for new work.

© Witmer

14 · WHERE DOES THIS FIT?

Where this sits in the Gym

If the question is...	Use this
“What must not be interrupted?” (before the cycle)	**Protect the Critical Path** — this tool. Proactive: protect the pacing work before the urgent stuff eats it.
“What are we waiting for?” (work already stuck)	**Dependency Snapshot** — reactive: trace what the stuck work is waiting on, to the source.
“Where is time disappearing?” (in one piece of work)	**Constraint Snapshot** — find the stall point where work sits waiting.
“The whole list is too big — what do we cut?”	**Execution Queue Control** — manage what gets in, before it ever stalls.

Rule of thumb: if work has already stalled, you need to find a blocker (Dependency Snapshot). If nothing’s stalled yet and you’re defending a deadline, you’re protecting a path (this tool). Protect the Critical Path asks *what must not be interrupted*; Dependency Snapshot asks *what are we waiting on*.

© Witmer