OneThing for Teams – Playbook

Debjit Biswas

17 September 2025

This document is a practical and opinionated guide for managing focused, outcome-driven product development. It defines a system for how teams should prioritize, execute, and complete work, centered on the belief that focus, clarity, and deliberate scope control are key to delivering real user value.

It serves as a handbook for both leadership and teams to align on how to operate, communicate, and ship successfully, using the "One Thing at a Time" principle as its core operating rule.

1 Operating Manual

The Three Pillars of Launch Success

Initiative Prioritization

Choose the most valuable work based on user impact, urgency, risk reduction, and alignment with core vision.

Initiative Pruning

Adapt when reality shifts. Reduce scope intentionally to deliver usable value within constraints.

Team Focus

Avoid multitasking. Each team owns one initiative, commits fully, and finishes cleanly before moving on.

Core Philosophy

Teams do only one thing at a time. That thing is chosen based on what delays will cost us most. Anyone who's part of the team, fully or partially, gives it their attention when needed. Otherwise, they rest. That's fine.

2 Team Mechanics

Multitasking is forbidden. Idleness is acceptable.

- Each **team** works on exactly **one initiative** at a time.
- A team is assigned to **only one product** during a cycle.
- A person cannot be part of multiple teams concurrently.
- Teams are **dynamically formed** based on needs.
- When the initiative is done or abandoned, the team disbands or picks the next initiative.

Communication & Updates

- Teams should share **progress updates** at set intervals (e.g., weekly).
- Updates can include:
  - Short written notes
  - Demo videos
  - Links to test/staging/live environments
- All updates should be shared in the system to ensure visibility.
- Teams are encouraged to have **face-to-face conversations** (online or offline) to ensure alignment.
- **Asynchronous communication** is supported and encouraged for preserving deep work and individual focus.

Shared Roles

- People in shared roles (e.g., finance, legal, ops, depending on the initiative) may be assigned to a team.
- If assigned, they must be available on demand.
- Outside of those needs, they may be idle or self-directed.

Team Size

Research and practice suggest that software teams work best when they consist of **3 to 7 people**. This size strikes a balance: small enough to minimize coordination overhead, yet large enough to include the skills needed to deliver an initiative end-to-end.

The exact count should reflect only those roles that are truly core to the initiative. Depending on the nature of the work, some functions may not be required day-to-day and can be treated as shared resources outside the team size consideration.

Support Team

There is always a dedicated support team in every cycle. Its role is to handle bugs and urgent issues so that initiative teams can stay focused.

- Membership in the support team **rotates each cycle**, ensuring the responsibility is shared fairly across the organization.
- The support team is **not a frontline help desk**. They don't deal directly with every customer complaint or request. Instead, they act as a buffer between customer-facing support staff and initiative teams.
- They work on issues that require changes in the product or codebase, addressing the most critical first.
- If a severe situation arises, other teams may temporarily join in, but support remains the first line of response.

This structure ensures that user-facing support gets timely help without disrupting initiative teams, preserving focus while keeping quality high.

3 Initiatives

An *Initiative* is a discrete chunk of work that delivers visible user impact or reduces risk in a meaningful way. An initiative should be scoped such that it can be completed within a single cycle, validated in use, and clearly tied to product value.

Initiative Prioritization

Initiatives are prioritized based on Cost of Delay (CoD). When possible, use the real CoD expressed in monetary terms. If that cannot be defined, use the following proxies—User Value, Time Criticality, and Risk Reduction, and sum them to create a relative estimate of CoD.

- **User Value**: How much impact will this have on end users?
- **Time Criticality**: Will delaying this result in lost opportunity or increased cost?
- **Risk Reduction**: Does this remove uncertainty or technical/business risk?
- **Effort**: Can it be completed within reasonable budget?
- **Is Core**: Is this part of the product's core promise or vision?

Initiative Prioritization Formula

All initiatives should be ranked to support decision-making, but core initiatives are treated as non-negotiable, they must be executed. However, even core initiatives may vary in urgency and value, so they are still scored for internal ordering.

Use the following formula for ranking initiatives:

$$\text{WSJF}^{1} = \frac{(\text{User Value} + \text{Time Criticality} + \text{Risk Reduction})}{\text{Effort}}$$

- User Value, Time Criticality, Risk Reduction, and Effort are scored on a Fibonacci scale (e.g., 1, 2, 3, 5, 8).
- Core initiatives are always selected, but this score still helps decide **which core to do first** and how to sequence non-core work.
- Higher scores indicate higher priority.

This formula helps force-rank initiatives based on what delivers the most impact for the least effort, while ensuring core initiatives are not missed.

Only prioritized initiatives are eligible for team assignment.

Initiative Properties

- Title/Description
- Product
- A **Time Budget** (max estimates)
- An optional common **chat room link**
- A **Progress Status**, one of: Thinking, Trying, Building, Finishing, Deploying, Stuck
- A **Lifecycle Status**: Waiting, Active, Done, Abandoned, Pruned
- Prioritization Properties (see above)
- Effort Estimate (Fibonacci scale)
- WSJF (automatically calculated)

[1] Weighted Shortest Job First (WSJF) is a prioritization model used to sequence work for maximum economic benefit. It is based on the idea that the most valuable work is that which delivers the greatest benefit in the shortest possible time. By dividing the combined impact (value, criticality, and risk reduction) by the effort required, WSJF ensures that teams focus on high-impact, low-effort items first.

- Assigned Team (if any)
- Subtasks (optional; each with its own priority)

No team can pick up a new initiative unless their current one is complete, pruned, or officially abandoned.

Initiatives cannot roll over to the next cycle by default. If a initiative cannot be completed within the current cycle, it must either be pruned to deliver something useful, or abandoned and re-prioritized for a future cycle.

The system may assist in pruning by tracking subtasks and their priorities. As time runs out, it can automatically suggest which lower-priority subtasks to drop in order to deliver the most valuable outcome within the remaining budget.

4 Cycles

Teams may choose any cycle length that suits their context. The important part is to have a fixed timebox that provides both focus and a natural point to reassess priorities.

This playbook suggests **six-week cycles**[2], inspired by the approach described in Basecamp's Shape Up methodology. Six weeks was found to be a sweet spot:

- **Long enough** to deliver something meaningful from start to finish.
- **Short enough** that the end feels visible from the beginning, which encourages trade-offs and avoids drifting.

Avoids the heavy overhead of very short cycles (e.g., two-week sprints) where planning and regrouping consume more time than the actual work accomplished.

After each cycle, a **two-week break/stabilization period** is recommended for fixing post-release issues, reviewing outcomes, preparing for the next cycle, or simply resting and prototyping.

5 Bugs

- All bugs are fixed immediately.
- No bug backlog is maintained.
- If a bug is deemed non-critical, it is ignored.

[2]`https://basecamp.com/shapeup/2.2-chapter-08#six-week-cycles`

Who fixes bugs?

- The support team for the current cycle handles all bugs.
- If multiple critical bugs arise, support team handles them sequentially.
- If the situation is severe, everyone drops everything to fix.

6 Done = In Use

An initiative is only marked done when users are using it.

Validation Before Done

- Before a initiative is marked done, the system should enforce validation:
  - A URL where the initiative is live
  - A tracking event or usage metric
  - A screenshot or evidence of deployment
  - Manual sign-off by a product/UX lead
- If validation is missing, the system should:
  - Warn that the initiative is not confirmed in use
  - Block marking it as done, or mark as pending validation
  - Allow manual override, but log who did it and why

No exceptions

- Merged but unused? Not done.
- Deployed behind a flag? Not done.
- Blocked by lack of communication? Not done.

7 Abandonment & Pruning

Abandonment

Stopping early is better than wasting effort.

- An initiative can be abandoned if continuing no longer makes sense.
- Abandonment must be a deliberate decision by product leadership.
- Teams may then disband or be reassigned.
- Abandoned initiatives are not considered failures if the decision is made in time.

Pruning

Pruning is a sign of focus, not failure.

- An initiative can be pruned when it becomes clear it cannot be completed in time.
- Pruning is the act of deliberately scaling back the scope of the initiative to ensure a partial but meaningful outcome.
- Pruned initiatives are still considered complete if the reduced version is delivered and used.
- Teams and leadership must agree on the pruned version and redefine "done" accordingly.

8 Emergencies

Emergencies are the only valid interruption.

- If something explodes, we drop everything.
- Everyone is expected to help fix.
- Emergencies are rare, but treated seriously.

9 Cultural Principles

- Focus is sacred.
- Prioritization is economic, not political.
- Idleness is better than context-switching.
- We optimize for flow, not busyness.
- Bugs are respected by fixing, not tracking.
- Done means impact, not code.
- Abandonment is a strength, not a weakness.
- Pruning is deliberate clarity, not cutting corners.
- Carrying over incomplete work is a silent failure. Decide clearly: prune, abandon, or finish.

10 Summary Checklist

- Before Starting a Initiative
  - ☐ Has it been ranked?
  - ☐ Do we have a full team with no conflicts?
  - ☐ Can we finish it within the time budget allocated?
- While Working
  - ☐ Are we ignoring everything else?
  - ☐ Are bugs being handled by support?
  - ☐ Are shared members showing up when needed?
- Before Marking Done
  - ☐ Are real users using it?
  - ☐ Is impact visible?
  - ☐ Has the system validated deployment through link, metric, or sign-off?
- If Considering Abandonment
  - ☐ Is the initiative still worth pursuing?
  - ☐ Has the cost of continuing exceeded its value?
  - ☐ Has leadership confirmed abandonment?
- If Considering Pruning
  - ☐ Is the original initiative now unrealistic within the time budget?
  - ☐ Can we redefine the scope meaningfully?
  - ☐ Will the pruned initiative still deliver user value?
  - ☐ Has leadership signed off on the new definition of done?
  - ☐ Has the system suggested a subtask pruning strategy based on remaining time?

11 Final Words

Do one thing. Do it right. Do it until users feel it.

Or stop if it no longer matters.

Or trim it if it still matters, but time has run out.