Binary Semantics®

STANDARD OPERATING PROCEDURE

Production Rollout Playbook

Four rollout types · Templates · PM effort reference

Owner	Aman · Product | FleetRobo
Date	12 May 2026
Version	1.0 (Draft)
Status	For review — PMO Lead, Product PMs, Eng Lead

INSIDE THIS DOCUMENT

DISTRIBUTION	CLASSIFICATION	NEXT REVIEW
Product · PMO · Eng · QA · CS	**Internal Use**	**2026-06-09**

A living document. Use it, break it, tell me where it breaks. v1.1 in 4 weeks.

How features actually reach clients — sized by item type, not tier

Owner: Aman · Product | FleetRobo **Version:** 1.0 (Draft) **Date:** 2026-05-12 **Companion to:** PMO ↔ Product ↔ Engineering SOP v1, Severity Tier System v1

---

0. Why This Exists

A Dev release is not a client release. The current pain is that the moment Dev says "done," features get pushed to production with no release notes, no training, no client warning — and the value of the work evaporates because clients can't see what changed.

This playbook fixes that with **four named rollout types**, each with:

- A clear definition of what fits in it
- A timeline from Dev release to client release
- A template for every artifact
- A default owner and effort estimate

The rule: **PM picks the rollout type at planning time, not at release time.** By the time Dev hands off, the rollout is already documented. No scrambling.

---

1. The Four Rollout Types

#	Type	Use For	Time from Dev Release to Client	PM Effort
1	**Direct**	Single-client bug fixes, server-side patches with no UI change	Same day	30 min
2	**Notified**	Cross-client bug fixes, single-client small changes, visible UX tweaks	1–2 days	1–2 hours
3	**Demoed**	New small features, cross-client UX changes, integration updates	3–5 days	Half day
4	**Piloted**	New modules, breaking changes, anything field-dependent, major features	1–3 weeks	1–3 days spread over rollout

Default mapping (PM can override at planning, but if blank, system picks):

Tracker `Type` field	Lane	Default Rollout
Bug (single client)	B (QA-routed)	Type 1 — Direct
Bug (cross-client)	B (QA-routed)	Type 2 — Notified
Small Change	A (PM-routed)	Type 2 or 3 — depends on visibility
Development / Enhancement	A (PM-routed)	Type 3 — Demoed
Major Feature / New Module	A (PM-routed)	Type 4 — Piloted
P0 Incident hotfix	C (incident)	Type 1 — Direct (post-mortem comms separate)

The default exists so PMs don't have to think hard for routine items. If something feels off about the default, bump it up one type. Over-prep is recoverable; under-prep is a client escalation.

Lane Constraint on Rollout Type

Lane B items can ONLY use Type 1 or Type 2. If a bug fix turns out to need Type 3 (Demoed) or Type 4 (Piloted), it **auto-promotes to Lane A** — because Type 3+4 require PM-owned artifacts (training material, sales kit, pilot agreement, retro) that QA Lead cannot produce alone.

Promotion triggers (Lane B → Lane A):

- Fix requires demo or training material → Type 3 → Lane A
- Fix is risky / field-dependent / breaking → Type 4 → Lane A
- Client comms beyond release notes are needed → Lane A
- Product judgment surfaces during scoping → Lane A

QA Lead initiates the promotion via tracker comment. PM is notified, can accept or ask for more detail.

---

2. Type 1 – Direct Rollout

Use when: The change is invisible to the user OR affects only one client OR is a critical fix that needs to be in production immediately.

Examples:

- Backend performance fix (no UI change)
- Single-client bug fix with workaround already in place
- Hotfix for a P0 incident after mitigation
- Server-side config correction

Timeline

```
TUE     Dev Release → internal env (build verified by PM)
TUE/WED Push to production (PM signs off)
TUE/WED 1-line release note sent to CS lead (Teams DM)
        Affected client(s) notified by CS (1-2 sentence email)
```

Total: same day or next morning.

Artifacts

Artifact	Owner	Format
Internal release note	PM	1 line in tracker (`Released: Fixed X for Y`)
CS notification	PM → CS	Teams DM or email
Client notification (if affected)	CS	1–2 sentence email

Template — CS Notification (Teams/Email)

```
Subject: Released — <ticket ID> — <one-line title>

Hi <CS Lead>,

<Ticket ID> is now in production.
Change: <one line on what was fixed>
Affected clients: <list or "internal-only">
No action needed from your end unless a client asks.

Tracker: <link to ticket>
```

When NOT to use Type 1

- The change is visible to multiple clients → use Type 2
- The fix introduces new behaviour clients should know about → use Type 2
- Anyone outside Eng/PM might need to demo it → use Type 3

---

3. Type 2 — Notified Rollout

Use when: The change is visible to clients but is contained — a fix or small UX change that doesn't require training, but clients should know it happened.

Examples:

- Cross-client bug fix (e.g. report column was wrong, now correct)

- Visible UX tweak (button moved, filter added)
- API behaviour change (backward-compatible)
- New field on existing report

Timeline

```
TUE     Dev Release → internal env (build verified by PM)
WED     Release note written (PM, ~1 paragraph)
WED     CS heads-up — 15-min walkthrough or async message
WED/THU Push to production
THU     Client comms email sent (CS executes, PM provides content)
```

Total: 1–2 days from Dev Release to client visibility.

Artifacts

Artifact	Owner	Format
Internal release note	PM	1 paragraph in tracker + Teams post
CS heads-up	PM → CS Lead	15-min walkthrough or async message
Client comms email	CS (content from PM)	Templated email per §3 below

Template — Release Note (Internal, Teams)

```
🚀 Released — <Ticket ID>

What changed: <2–3 sentences>
Affected clients: <list>
Where to see it: <module / screen / API endpoint>
Known limitations: <if any, or "none">

Tracker: <link>
```

Template — Client Comms Email

```
Subject: <Module> update — <one-line summary>

Hi <Client Name>,

We've shipped an update to <module> that <one-line change>.

What's new for you:
• <bullet 1>
• <bullet 2>

Where to find it: <link or path>

If you spot anything unexpected, reply to this email or contact <CS contact>.

— <CS contact>
   Binary Semantics · FleetRobo
```

When NOT to use Type 2

- The change requires training to use → upgrade to Type 3
- Multiple stakeholders need a demo → upgrade to Type 3
- It's a new feature, not a fix → upgrade to Type 3

---

4. Type 3 — Demoed Rollout

Use when: A new feature or significant enhancement is shipping. Clients won't get full value without seeing how it works.

Examples:

- New filter or report type
- New module surface (e.g. a new dashboard view)
- Integration with a new third-party tool
- Workflow change that affects daily operations

Timeline

```
TUE     Dev Release → internal env (build verified by PM)
WED     Release note written (PM)
WED     Training material drafted (PM + CS)
WED     PMO + CS demo (30 min, PM presents)
THU     Sales informed if revenue-relevant (Sales kit, if needed)
THU     Client comms drafted by CS, PM reviews
FRI     Push to production
FRI     Client comms email sent
NEXT WK CS reaches out for client demo if requested
```

Total: 3–5 days from Dev Release to client release.

Artifacts

Artifact	Owner	Format
Release note	PM	1 page, includes screenshots
Training material	PM + CS	1–2 pager OR 5-10 min video
Internal demo	PM	30-min Teams call with PMO + CS
Sales kit (if revenue-relevant)	PM	1-pager: what it is, who it's for, talking points
Client comms email	CS (content from PM)	Templated email per below
Optional client demo	CS	15-30 min on request

Template — Release Note (Client-Facing)

```
<Feature Name>
Released: <date>

What is it?
<2–3 sentences explaining the feature>

Who benefits?
<roles / use cases>

How to use it:
1. <step>
2. <step>
3. <step>

Screenshots: <link or inline>

Known limitations:
<list if any>

Questions? Contact <CS contact>.
```

Template — Sales Kit (1-pager)

```
<Feature Name> — Sales Brief
Released: <date>

The pitch (1 sentence):
<one-line value prop>

The numbers:
• <metric 1 if known, e.g. "saves 20 min/day for ops users">
• <metric 2>

Who to pitch to:
• <persona / role>
• <client type>

Talking points:
1. <point>
2. <point>
3. <point>

Objection handling:
• "<common objection>" → <response>

Demo flow (3 min):
Step 1: <click X>
Step 2: <show Y>
Step 3: <highlight Z>
```

Template — Client Comms Email (Type 3)

```
Subject: New in FleetRobo — <Feature Name>

Hi <Client Name>,

We've shipped <Feature Name> — <one-line value prop>.

What it does:
<2-3 sentences>

How it helps your team:
• <bullet>
• <bullet>

You can find it under <location>.

Want a 15-min walkthrough? Reply to this email and we'll set it up.

— <CS contact>
   Binary Semantics · FleetRobo
```

When NOT to use Type 3

- The change is risky to roll out broadly → upgrade to Type 4 (pilot first)
- A specific client must validate before broad rollout → upgrade to Type 4
- The feature requires field testing or hardware coordination → upgrade to Type 4

---

5. Type 4 — Piloted Rollout

Use when: The change is significant, risky, or untested in real-world conditions. We deliberately ship to one or a small set of clients first, then expand.

Examples:

- New module (e.g. a completely new dashboard, new vehicle category support)
- Breaking change to an existing workflow
- Anything field-dependent (vehicle behaviour, hardware coordination, device firmware)
- Major API changes
- Cross-product changes (e.g. LPA ↔ TMS integration)

Timeline

```
WEEK 1
TUE     Dev Release → internal env
TUE–WED PM verifies build + internal demo (PMO + CS + Sales)
WED–THU Training material + sales kit + demo video produced
THU–FRI Pilot client confirmed (CS Lead nominates, PM approves)
FRI     Pilot client briefed + go-live agreed

WEEK 2
MON     Pilot client receives feature (live in production for them only)
MON–FRI Daily check-ins with pilot client (CS + PM)
FRI     Pilot retro — issues, feedback, blast-radius confirmation

WEEK 3
MON     Phased rollout begins (tier-1 clients first if applicable)
WED     Mid-rollout health check (monitoring, escalations)
FRI     Full client base released

WEEK 4
MON–FRI Stabilisation window — monitoring, support, rapid fix cycle
FRI     Post-rollout review (PM + Eng + CS)
```

Total: 1–3 weeks from Dev Release to full client release.

Artifacts

Artifact	Owner	Format
Release note	PM	Full doc (3–5 pages with screenshots)
Training material	PM + CS	10–15 slide deck OR 10–20 min video
Demo video	PM	3–5 min screen recording
Internal demo	PM	30–60 min Teams call (PMO + CS + Sales + Eng Lead)
Sales kit	PM + Sales	Full deck for client-facing pitch
Pilot client written agreement	CS Lead	"This is a controlled rollout, here's the support plan"
Daily check-in notes	CS + PM	Tracker comments during pilot week
Pilot retro doc	PM	1-pager: what we learned, what we changed, go/no-go
Phased rollout schedule	PM	Calendar of which clients get it when
Rollback plan	Eng Lead	Documented + tested before pilot starts
Post-rollout review	PM + Eng + CS	30-min retro at end of stabilisation window

Template — Pilot Client Agreement (CS sends to client)

```
Subject: Early access — <Feature Name> — controlled rollout

Hi <Client Name>,

We're shipping <Feature Name> and we'd like <Client Name> to be one of the first to us

What this means:
• You get the feature 1–2 weeks before the broader client base
• You get direct access to our PM team during the rollout week
• We expect feedback — what's working, what's broken
• If we find issues, we may need to pause or rollback for you while we fix

Your commitment:
• Use the feature in your normal operations
• Report any issues to <CS contact> within 24 hours
• Join a 15-min check-in call mid-week

Our commitment:
• Daily monitoring for the first 5 days
• Same-day response on any issue you raise
• Rollback within 1 hour if you ask
• Full release notes + training material before go-live

Are you in? Reply to confirm and we'll schedule the go-live.

— <CS contact>
   Binary Semantics · FleetRobo
```

Template — Pilot Retro (PM writes after pilot week)

```
<Feature Name> — Pilot Retrospective
Pilot client: <name>
Pilot duration: <dates>

What happened
<3–4 sentences summary>

Issues found
1. <issue> — <severity> — <resolution>
2. <issue> — <severity> — <resolution>

Pilot client feedback
<1–2 paragraphs of what they said>

Decision
□ GO — proceed to phased rollout as planned
□ HOLD — fix [issues] first, re-pilot
□ NO-GO — rework needed, return to Dev

Phased rollout schedule
Week 1: <client list>
Week 2: <client list>
Week 3: full rollout

Rollback trigger
We will rollback if: <criteria>
```

When NOT to use Type 4

- The change is small and contained → use Type 2 or 3 instead
- Time pressure is severe and risk is well-understood → use Type 3 with extra monitoring
- The pilot would delay a critical business need → escalate to Head of Product for override

---

6. The Rollout Decision Flow

A one-screen decision tree for picking the right type at planning time.

```
START
  |
  ▼
Is this a bug fix?
  |
  ├── YES ──→ Affects multiple clients OR visible UX change?
  |             ├── NO  ──→ TYPE 1 (Direct)
  |             └── YES ──→ TYPE 2 (Notified)
  |
  └── NO ──→ Is it a new feature or significant enhancement?
               |
               ├── NO  ──→ TYPE 2 (Notified) — UX tweak or contained change
               |
               └── YES ──→ Is it risky, field-dependent, or breaking?
                             ├── NO  ──→ TYPE 3 (Demoed)
                             └── YES ──→ TYPE 4 (Piloted)
```

Three sanity checks before locking the type:

1. **Could a confused client cost us business?** → Bump up at least one type.
2. **Could rolling back take >1 hour?** → Use Type 4 (pilot first).
3. **Could a single client's feedback save us from breaking 10 others?** → Use Type 4.

---

7. PM Effort Reference

How much work does each rollout type really demand from PM?

Type	Total PM Time	Spread Over	What PM actually does
1 Direct	30 min	Same day	1 Teams DM + 1 tracker comment
2 Notified	1–2 hours	1–2 days	Release note + heads-up to CS + email content
3 Demoed	Half day	3–5 days	Above + training material + internal demo + sales kit if needed
4 Piloted	1–3 days spread	1–3 weeks	Above + pilot agreement + daily check-ins + retro + phased rollout management

Cognitive load rule: The tracker tells PM what to do next. Status flows: `Planning → Decided → In Dev → Dev Released → Rollout Type Confirmed → In Rollout → Released`. PM follows the status, doesn't have to remember the steps.

Templates do the writing. For every artifact above, there's a template (this doc + the SOP). PM fills blanks; doesn't write from scratch.

CS owns comms execution. PM provides content (1–3 sentences); CS sends the client emails. PM never writes 5 client emails individually.

---

8. The Tuesday Sync – What PM ↔ PMO Discusses

The weekly Tuesday PM ↔ PMO sync is the heartbeat. Items by phase:

Phase 1 (Planning):

- Any new tickets received since last Tuesday?
- Anything escalated this week (client pressure, sales motion)?
- What's been responded to / decided / parked?

Phase 2 (Dev Build):

- What does Dev have in their current sprint?
- Any blockers or slipping items?
- Anything finishing this week (heading to Dev Release on Tuesday)?

Phase 3 (Rollout):

- What was Dev-released this Tuesday?
- What rollout type was assigned?
- What's the client comms plan?
- Any items in stabilisation (post-rollout)?

Out-of-band escalation: Discussion can start anytime. Tuesday is the minimum cadence, not the only window. P0/P1 items get same-day conversations.

---

9. Friday – Decisions Day

Friday is when the response window closes for everything that arrived during the week. Every open ticket has ONE of three outcomes communicated:

Decision	What it means	What PM communicates
Accept	Going into upcoming Dev sprint	Sprint date + target Dev Release date + rollout type
Decline	Not happening (duplicate, out of scope, won't fix)	Reason + suggested alternative if any
Park	Accepted but not now	Revisit date + dependency / trigger (e.g. "after migration X", "next quarter planning")

By Friday EOD, every reporter knows where their ticket stands. This kills the "ticket black hole" problem where PMO submits something and hears nothing for 3 weeks.

Communication channel:

- Tracker status updated (single source of truth)
- Teams DM to reporter (synchronous heads-up)
- Decisions log in the Tracker (separate sheet/tab for traceability)

---

10. Rollout Type Examples — Walked Through

Three real examples from the current VT live tracker, showing how rollout type would be picked.

Example A — VT-001 (Trip Creation Failure, P0 Incident)

Rollout Type: 1 (Direct) if mitigated server-side without UI change.

Reasoning: P0 hotfix. Get it to production as fast as possible. Comms separate from rollout (post-mortem RCA goes to affected clients within 72 hours per the SOP).

Example B — VT-011 (Real-Time Alerts on WhatsApp, P1 Development)

Rollout Type: 3 (Demoed)

Reasoning: New surface (WhatsApp delivery) requires CS to understand the configuration. Two POC clients waiting (Adani Cement, Adani Hazira). Sales kit useful — this is a competitive feature.

Timeline:

- Dev Release Tue → release note + training mat Wed → PMO+CS demo Wed → Sales brief Thu → Push Fri → Client comms Fri

Example C — VT-014 (Live Stream Latency, Stability Track)

Rollout Type: 4 (Piloted) when fix is ready.

Reasoning: Cross-port impact, recurring issue, fix likely to be infra-level. We don't want to fix something for one client and break another. Pilot with Adani Karaikal (already affected, will validate quickly), then phased rollout to other ports.

Note: Stability Track items follow the same rollout playbook as features once a fix is ready — the *track* dictates how it's planned, the *rollout type* dictates how it ships.

---

11. Metrics

Tracked monthly. Rollout-discipline metrics, separate from delivery metrics.

Metric	Target	Why
% of releases with rollout type assigned at planning (not at release)	≥ 90%	Discipline marker — no scrambling at release time
% of Type 3+ releases shipped with full artifact set	≥ 95%	Artifact gate working
% of Type 4 releases that did a pilot	100%	The pilot is non-negotiable
Avg time from Dev Release to Client Release — Type 1	≤ 1 day	Sanity check
Avg time from Dev Release to Client Release — Type 2	≤ 2 days	Sanity check
Avg time from Dev Release to Client Release — Type 3	≤ 5 days	Sanity check
Avg time from Dev Release to Client Release — Type 4	1–3 weeks	Sanity check
Post-release client escalations within 7 days	Trending down	Quality marker
Releases requiring rollback	< 5% of releases	Pre-release validation working

12. Open Items

1. **Sales kit ownership** — is Sales producing their own slide layer, or does PM hand them a ready-to-use deck? Confirm at first usage.
2. **Pilot client roster** — CS Lead to maintain a list of friendly, low-risk pilot candidates per product (FleetRobo, TMS, VT, LPA). Refreshed quarterly.
3. **Rollback testing** — for Type 4 rollouts, is rollback actually tested in staging before pilot, or only documented? Eng Lead to confirm.
4. **Demo video toolchain** — what does PM use to record demos? Loom, in-house tool, screen recorder? Standardise to reduce friction.
5. **Phased rollout client tiering** — is there an existing "tier-1 vs tier-2 client" classification, or do we create one? Affects Type 4 phasing logic.
6. **Decisions log format** — Friday Accept/Decline/Park log lives where? Separate tab in tracker, separate doc, Notion page? Pick one.

This playbook is a living document. PMs and PMO should challenge any rollout-type assignment that feels wrong. v1.1 after first month of real use.

— Aman Product | FleetRobo