

SaaS Development Operations Playbook

GCom Pro | Version 1.0 | October 5, 2026

Purpose. Use the SaaS Development dashboard to coordinate fictional product, engineering, reliability, implementation and support work. Five SOPs and four templates connect visible task and budget signals to an accountable next action and recorded acceptance.

Demonstration scope. All tasks, roles, amounts and examples are invented. These procedures describe an illustrative operating model. They do not deploy software, monitor a service, approve a production change or communicate with actual customers.

Use this pack. Start with the metric dictionary. Choose the procedure matching the issue, identify the responsible role and record the decision and evidence. Inspect the task in Source data. Keep release identifiers, test results, incident timelines and acceptance records in a separate mock delivery log when the dashboard has no field for them.

Current capabilities. The dashboard shows task status, workstreams, due dates, budget, actual spend and missing fields. It does not calculate velocity, deployment frequency, change failure rate, uptime, latency, engineering capacity or product adoption. A task named for a release or incident is a coordination record, not telemetry.

Document control. Mock owner: Delivery Operations Lead. Review quarterly and when source fields, formulas or operating rules change. Record the reviewer, version and acceptance date. Publishing this pack does not configure new approval workflows, monitoring alerts or release pipelines.

Contents. Operating roles and metric dictionary; SOP 01 Backlog readiness and acceptance; SOP 02 Release readiness and decision; SOP 03 Defect and incident coordination; SOP 04 Implementation and support handoff; SOP 05 Delivery cost and scope review; four reusable templates.

Operating Roles and Metric Dictionary

Use delivery records to distinguish administrative progress from software and service outcomes.

Daily review. Delivery Operations Lead checks source freshness, Stuck tasks, overdue open work and missing fields. Select up to three priorities with a role, next action and date. Weekly, reconcile release dependencies, implementation handoffs and cost exceptions.

Mock roles. Product Owner defines the intended outcome and acceptance criteria. Engineering Lead owns technical review and dependencies. QA Reviewer records test evidence. Incident Coordinator owns the mock incident timeline. Implementation Lead and Support Lead accept operational handoffs. Finance Reviewer checks cost assumptions. A sender retains ownership until the receiving role accepts.

Task measures. Open work = total DEMO | tasks minus Done. Completion = Done / total, rounded. Needs attention counts Stuck tasks only; it does not include every overdue item or service risk. Task completion does not establish feature adoption, release success or incident recovery.

Budget measures. Reported totals sum available amounts and disclose missing values. Comparable utilization = matched spend / matched budget using only rows with both amounts. Remaining matched budget = matched budget minus matched spend. Missing is not zero. With zero matched budget, the current display falls back to 0%; that is not meaningful utilization.

Charts. Workstream progress shows the three largest workstreams by task count. Cost comparison uses the three largest by reported budget. Upcoming work groups task due dates into six weeks from Monday of the current UTC week, including Done tasks. Undated and out-of-window tasks are excluded. This is not a historical delivery trend or sprint capacity forecast.

Completeness and controls. Required fields are Status, Workstream, Next Action, Due, Budget and Actual Spend. Completeness does not establish test success or approval. Use only approved DEMO | records; preserve 20–30 unarchived tasks, at least six open and 2–4 labeled demonstration gaps. Never reopen Done work or replace unknown amounts with zero. Inspect cycle markers, read back changes and verify synchronization.

Separate evidence. Commit, build, test, deployment and incident-event records are needed for engineering performance measures. Story points, estimates, available hours and consistent scope are needed for capacity or velocity analysis. These are proposed additions, not current dashboard calculations.

SOP 01 Backlog Readiness and Acceptance

Owner: Product Owner. Technical review: Engineering Lead. Acceptance: named mock reviewer.

Trigger and inputs. A fictional request enters the delivery queue or its scope changes. Inputs: intended user outcome, affected component, dependency list, acceptance criteria and source task reference.

1 Define the outcome. Describe the sample problem, intended user and observable result. Identify what is included and excluded. Reuse an existing DEMO | task for the same open outcome rather than creating a duplicate.

2 Establish readiness. Record the Product Owner, Engineering Lead, dependencies and testable acceptance criteria in the mock log. Keep delivery estimates separate from the task Due field. Label unknown evidence instead of treating a filled checklist as proof.

3 Resolve dependencies. Name the decision or input needed and its owner. Use Stuck when a dependency prevents the task from progressing. A due date or cost allocation alone does not make work ready for implementation.

4 Review the result. The named reviewer compares mock output and test evidence with the agreed criteria. Record Accepted, Rework or Deferred and the reason. Scope changes require revised criteria and an explicit decision, not silent acceptance.

5 Close the coordination task. Mark Done only after the intended administrative outcome is accepted and all six required fields are present. Preserve unresolved follow-up work as a separate owned task. Completed history stays closed if a new issue appears.

Exception. Ambiguous criteria or missing evidence keeps acceptance provisional. Do not infer tested functionality from task completion percentage.

Completion evidence. Outcome and scope, criteria version, reviewed mock result, acceptance decision, reviewer and remaining actions.

Dashboard connection. Product and Engineering workstream progress, Stuck count, due-date distribution and task completeness. Story-point velocity and engineering capacity are not calculated.

SOP 02 Release Readiness and Decision

Owner: Delivery Operations Lead. Decision: designated mock release approver.

Trigger and inputs. A fictional release reaches its readiness review. Inputs: release identifier, included scope, test summary, unresolved defects, dependency status, rollback outline and support handoff record.

1 Identify the release scope. List the sample features, changes and dependencies covered by the review. Link relevant tasks and record the scope version. Distinguish a completed readiness task from an actual deployment event.

2 Review evidence. Engineering Lead and QA Reviewer summarize mock checks, known issues and residual risks. Record evidence references and outcomes. Missing tests remain missing; they cannot be recorded as passed.

3 Check operational readiness. Confirm the mock owner, proposed window, rollback decision criteria, observation plan and support knowledge handoff. Record which conditions remain unresolved and who owns them.

4 Record the decision. The designated approver records Go, Conditional or Hold with rationale. A blocking dependency keeps the coordination task Stuck or open as appropriate. No dashboard status change triggers a real release.

5 Verify the handoff. Record acceptance by the receiving operations or support role. Mark the readiness task Done only when its agreed review outcome and required fields are complete. Deployment execution and post-release verification remain separate records.

Exception. Missing approval, rollback evidence or critical acceptance results keeps the simulated release decision on Hold. A high completion percentage does not override a release condition.

Completion evidence. Release scope and version, evidence summary, known issues, recorded decision, receiving-role acceptance and owned conditions.

Dashboard connection. Engineering and Support tasks, release-preparation budgets and due dates. Deployment frequency and change failure rate require actual deployment-event populations and outcome records; they are proposed metrics.

SOP 03 Defect and Incident Coordination

Owner: Incident Coordinator. Technical review: Engineering Lead. All incidents are simulated.

Trigger and inputs. A fictional defect, degraded-service report or reliability dependency needs coordination. Inputs: sample symptoms, affected scope, mock impact, evidence reference and related task.

1 Record the issue. State the observed mock symptom and affected sample users or component. Separate a defect report from an active service incident. Record an illustrative severity and rationale in the log; the dashboard has no severity timer.

2 Assign coordination. Name the Incident Coordinator, technical reviewer and next evidence-gathering action. Use Stuck for a blocking dependency. Do not imply that a synthetic task is an actual monitoring alert.

3 Maintain an evidence timeline. Record simulated detection, investigation, mitigation and validation events with consistent timestamps. Keep observations, assumptions and decisions distinct. A task Due date is a coordination deadline, not a recovery timestamp.

4 Validate the mock outcome. The technical reviewer records what was checked, the result and any limitation. The receiving role acknowledges the simulated recovery or defect disposition. Draft any communication in the exercise log; send nothing to real customers.

5 Close and follow through. Mark the coordination task Done only after its outcome is accepted and all required fields are complete. Keep prevention, documentation or remediation follow-ups separately owned. Create a new eligible DEMO | record for a recurrence; do not reopen completed history.

Exception. Unverified recovery remains unresolved. Missing incident-event evidence cannot be substituted with a task status or a zero-duration result.

Completion evidence. Issue reference, impact rationale, event timeline, reviewed mock result, acceptance and follow-up owner/date.

Dashboard connection. Reliability work, Stuck tasks and missing evidence. Availability, response time and recovery duration need service telemetry and defined event boundaries; no SLA attainment or recovery clock is currently calculated.

SOP 04 Implementation and Support Handoff

Owner: Implementation Lead. Receiving role: Support Lead. Acceptance: named mock business reviewer.

Trigger and inputs. A fictional implementation, integration or knowledge update is ready for handoff. Inputs: sample tenant scope, agreed outcomes, mock validation, access assumptions, known issues and support instructions.

1 Define the handoff. Record what the sample implementation delivers, which integrations are included and which support role receives it. Use invented tenant and account references only. Do not store actual credentials or customer information.

2 Verify readiness. Review the mock configuration and integration results against accepted scope. Document unresolved exceptions and their effects. Confirm that the knowledge article or training reference matches the reviewed version.

3 Review with the receiver. Support Lead checks the handoff record, troubleshooting reference, escalation owner and known limitations. Record Accepted or Returned with a reason. The sender keeps ownership until acceptance is explicit.

4 Separate delivery from adoption. Record an observed mock first-value result only when evidence supports it. Training completion, handoff acceptance and product adoption are different outcomes. Task completion does not automatically update Customer Success account measures.

5 Reconcile task records. Update the next action, supported spend and agreed due date. Mark Done only after the administrative handoff is accepted and required fields are present. Keep remaining implementation or knowledge gaps open and owned.

Exception. Missing receiving-role acceptance or unverified integration evidence prevents closure. A completed training task alone does not prove the sample customer has achieved value.

Completion evidence. Scope/version, validation record, known issues, knowledge reference, receiving-role acknowledgment and remaining actions.

Dashboard connection. Implementation and Support workstream status, budgets and missing fields. Account health, adoption and first-value measures belong to the separate Customer Success account model, not this task tab.

SOP 05 Delivery Cost and Scope Review

Owner: Delivery Operations Lead. Cost review: Finance Reviewer. Scope decision: Product Owner.

Trigger and inputs. A weekly delivery review, scope change or budget exception needs a decision. Inputs: task budgets, recorded spend, covered work and period, mock cost evidence and separately documented remaining-work estimates.

1 Establish a comparable basis. Confirm which fictional work and cost categories each amount covers. Distinguish labor, vendors and infrastructure costs where relevant. Keep committed but unincurred costs separate from actual spend.

2 Reconcile evidence. Compare available amounts with the mock cost record. Identify missing budget or spend values. Use only rows with both amounts for comparable utilization and remaining matched budget. Do not fill an unknown amount with zero.

3 Explain scope and variance. Calculate budget less spend and state what work remains. An underspend is not proof of efficiency when scope is unfinished or cost evidence is missing. Record changes to scope and their approval assumptions.

4 Review the forward view. If a separate remaining-cost estimate exists, add it to spend to calculate a mock forecast. Record assumptions and uncertainty. The dashboard does not currently store estimates to complete, engineering hours or forecast margin.

5 Record an owned decision. Product Owner and Delivery Operations Lead record a mock action: clarify scope, sequence work, request evidence or revise an allocation. State the owner and next review. Read back any task changes and preserve the original decision history in the log.

Exception. Incomplete evidence supports only a partial cost view. Zero matched budget has no meaningful utilization ratio. Task counts cannot establish available engineering capacity.

Completion evidence. Cost basis, reconciled amounts, missing-data disclosure, scope rationale, decision, owner and review date.

Dashboard connection. Reported budget/spend, matched utilization and workstream costs. Forecast cost, project margin and capacity measures remain separate proposed analyses.

Templates for Backlog and Release Reviews

Copy these structures into a fictional delivery log and retain the evidence references.

Backlog readiness and acceptance. Task/reference: ____ | Intended outcome: ____ | Included/excluded scope: ____ | Product Owner: ____ | Engineering reviewer: ____ | Acceptance criteria/version: ____ | Dependencies: ____ | Evidence: ____ | Accepted/Rework/Deferred: ____ | Next owner/action/date: ____

Backlog example. DEMO | Usage export acceptance: the sample export includes the requested columns, but the mock reviewer finds a missing date-boundary case. The task remains open for that evidence; overall completion percentage does not establish acceptance.

Release decision record. Release/reference: ____ | Scope/version: ____ | Test evidence: ____ | Open issues: ____ | Operational and support readiness: ____ | Rollback decision criteria: ____ | Go/Conditional/Hold: ____ | Approver and rationale: ____ | Receiving role: ____ | Conditions/owner/date: ____

Release example. A sample release has all feature tasks complete but its support handoff is unaccepted. The mock release decision remains Hold until that dependency is resolved. A readiness review may conclude with a documented Hold; it does not establish that software was deployed.

Proposed delivery measure. A separate mock release log records 10 production deployments and 2 that meet its documented failure definition: change failure rate is 20% for that period. The dashboard task table does not provide this denominator or classify deployment outcomes. Keep this illustrative calculation outside current task KPIs.

Templates for Handoffs and Cost Reviews

All values are invented. Preserve missing evidence and distinguish observed outcomes from assumptions.

Incident or implementation handoff. Record/type: ____ | Sample scope and impact: ____ | Sending owner: ____ | Receiving owner: ____ | Evidence/timeline: ____ | Validation result: ____ | Known limitations: ____ | Knowledge reference: ____ | Accepted/Returned and reason: ____ | Remaining owner/action/date: ____

Handoff example. A fictional API integration passes its mock payload checks, but support instructions omit the escalation owner. Support returns the handoff. The sender retains ownership until the missing instruction is supplied and accepted. No adoption or first-value result is inferred.

Delivery cost worksheet. Task/workstream: ____ | Covered scope and period: ____ | Budget: ____ | Recorded spend: ____ | Matched remaining budget: ____ | Separate remaining-cost estimate: ____ | Forecast total: ____ | Missing evidence: ____ | Scope decision: ____ | Owner and review date: ____

Matched-cost example. Task A has \$50,000 budget and \$32,000 spend. Task B has \$20,000 budget and missing spend. Reported budget is \$70,000; reported spend is \$32,000 with one missing-spend task. Comparable utilization is $\$32,000/\$50,000 = 64\%$, and matched remaining budget is \$18,000. Task B is excluded from the matched denominator.

Forecast example. A separate mock estimate assigns \$22,000 remaining cost to Task A. Forecast total is \$54,000 and forecast overrun is \$4,000. Remaining budget alone does not supply that estimate, and task completion does not determine remaining cost.

Logic review checklist. Trace each metric to its fields and formula; check completed and missing-value examples; verify date boundaries and denominators; distinguish task closure from release, recovery or customer acceptance; read back changes and confirm the published snapshot before reporting success.