OGO™ - OPRYNTA™ GUIDE OPS

User & Team Operating Playbook

DOCUMENTATION EDITION / FROM REPORTED PAIN TO PROVEN TEAM ACTION

A searchable operating knowledge base for technical account managers, solutions engineers, field engineers, platform and SRE teams, domain specialists, incident leaders, engineering leaders, and contributors.

LIVING SOURCE
/docs

REPOSITORY SOURCE
docs/*.md

EDITION
3.7.1

PUBLISHED
July 24, 2026

PUBLIC PRODUCT BOUNDARY

This edition documents the public OGO product: local-first guided diagnostic simulation, evidence gates, approved guide logic, owner routing, command purpose and safety, dossiers, validation, and contribution. It does not expose private Oprynta platform engines or unreleased operating-intelligence methods.

Contents

The hosted site supports full-text search, role filtering, chapter links, and on-page navigation. This publication preserves the same chapter sequence for offline use.

CHAPTER	TITLE	PRIMARY USE
01	**OGO documentation overview**	Start Here
02	**What OGO is and is not**	Start Here
03	**Quick start**	Start Here
04	**The shared operating path**	Start Here
05	**Incident operating rhythm**	Start Here
06	**Technical Account Manager playbook**	Roles
07	**Solutions Engineer and Architect playbook**	Roles
08	**Resident and Field Engineer playbook**	Roles
09	**Platform, SRE, and Operations playbook**	Roles
10	**Domain Specialist playbook**	Roles
11	**Incident and Program Leader playbook**	Roles
12	**Engineering and Operations Leader playbook**	Roles
13	**Technology domain walkthroughs**	Operating Safely
14	**Evidence quality and command safety**	Operating Safely
15	**Optional AI, BYOK, and privacy**	Operating Safely
16	**Dossiers and escalation packets**	Operating Safely
17	**Contributing diagnostic guides**	Build Contribute
18	**JSON guide authoring reference**	Build Contribute
19	**Validation, testing, and review quality**	Build Contribute
20	**Adoption and governance**	Build Contribute

21	**Workshop and tabletop facilitation**	Build Contribute
22	**FAQ and glossary**	Start Here
23	**Offline publication library**	Publications

CHAPTER 01 / START HERE

OGO documentation overview

Use the hosted knowledge base as the living source and the offline publications as versioned field editions.

ROLE PATHS	KEYWORDS	UPDATED
everyone	overview	2026-07-24

The publication model

OGO documentation is deliberately maintained in three connected forms. The hosted site is the fastest place to search, filter, and share a precise chapter. The repository Markdown is the reviewable source used by maintainers and contributors. The PDF and DOCX editions are versioned snapshots for workshops, offline use, procurement records, and structured review.

> **The website is the living source. The repository is the review surface. The publication files are controlled snapshots.**

This separation prevents a polished PDF from becoming a hidden knowledge silo. A user can search for a role, safety rule, evidence standard, or dossier requirement without opening a long file and scanning page by page.

How to use this library

1. Begin with Quick start if you are opening OGO for the first time.
2. Choose a role path when you need responsibilities, outputs, and handoff expectations.
3. Use Operate safely before running commands, handling customer evidence, or using optional AI.
4. Use Build and contribute when writing or reviewing diagnostic guides.
5. Download the PDF or DOCX when a fixed edition is required.

What belongs here

The library covers the public OGO product: local-first guided diagnostic simulation, evidence gates, approved guide logic, owner routing, command purpose and safety, dossiers, validation, and open-source contribution. It does not document private Oprynta platform engines, internal enterprise methods, unreleased workflows, or proprietary operating intelligence.

Search behavior

Search indexes titles, descriptions, headings, role tags, keywords, and article text. Press / from any documentation page to focus the search field. Filter by category or role when a broad query produces too many results. Each chapter has a permanent URL so a team can link directly to the exact operating guidance needed.

CHAPTER 02 / START HERE

What OGO is and is not

Understand the product boundary, the evidence-led operating model, and the responsibilities that remain human.

ROLE PATHS	KEYWORDS	UPDATED
everyone	product boundary	2026-07-24

What OGO does

OGO - Oprynta Guide Ops - is a local-first guided diagnostic simulator and operating playbook. It helps a team turn reported technical pain into a visible sequence of evidence questions, candidate fault boundaries, owner routes, safety gates, and measurable validation. The system does not replace monitoring, ticketing, engineering judgment, or formal change control. It creates a shared path through them.

The useful unit is not a generic answer. It is an evidence state: what is known, what remains unresolved, which question is next, who owns the answer, and what action is safe only after the boundary is supported.

What OGO does not do

- It does not claim autonomous root cause.
- It does not execute uncontrolled remediation.
- It does not hide command purpose or access requirements.
- It does not turn optional AI output into proof.
- It does not make every team perform the same work.

The operating promise

A good OGO run leaves the team with a decision-ready record. The record should distinguish symptoms from supported boundaries, observed evidence from assumptions, investigation from authorized change, and temporary recovery from verified return to service.

Human responsibility

Humans remain responsible for access, authorization, interpretation, change approval, escalation, and closure. OGO can organize the path and preserve the evidence, but the operator and accountable owner decide whether a command may run and whether validation is sufficient.

Quick start

Run a first local incident path from reported pain through evidence, owner routing, and validation.

ROLE PATHS	KEYWORDS	UPDATED
everyone	quick start	2026-07-24

Before the first run

Confirm that the repository dependencies are installed, the playbook validator passes, and the Studio opens locally. Optional AI is not required for the deterministic simulator. Start with a known scenario so the team can learn the operating rhythm before using an unfamiliar incident.

```
npm install
npm run validate:playbooks
npm run typecheck
npm run test
npm run dev
```

Start with a precise pain statement

Describe what broke, who or what is affected, when it began, whether a change correlates, what proof already exists, and what must not happen. A strong intake is observable and bounded. "The platform is slow" is weak. "Interactive jobs on one queue exceed the normal launch time after the scheduler policy update; batch jobs on the peer queue remain normal" is usable.

Follow the evidence gates

Answer only what can be supported. Use unknown when the evidence is not available. Capture timestamps, healthy-peer comparisons, command results, and physical observations. Do not select the answer that merely sounds most likely.

Review the tunnel

The compact Live Fault Tunnel shows current progress. Expand it to see the full topology: reported pain, committed evidence, current gate, candidate boundaries, owner route, safety gate, guided move, and exit proof. Close the map or press Escape to return without losing state.

Produce the record

At the end of the run, review the resolution, command record, owner handoff, escalation packet, and exit criteria. Export a dossier when another team, vendor, or reviewer needs a stable evidence package.

CHAPTER 04 / START HERE

The shared operating path

Use one evidence path across technical and organizational boundaries without collapsing distinct responsibilities.

ROLE PATHS	KEYWORDS	UPDATED
everyone	operating path	2026-07-24

The seven operating moves

6. Capture reported pain in observable terms.
7. Define affected scope and a healthy comparison.
8. Commit evidence at each gate without guessing.
9. Narrow candidate fault boundaries.
10. Route the next question or action to an accountable owner.
11. Separate investigation from authorized controlled change.
12. Prove the exit condition and preserve the record.

Why the path is shared

Incidents cross teams because technical layers and ownership layers do not align perfectly. A customer-facing lead may own impact and communication, a field engineer may own local proof, an SRE may own production safety, and a specialist may interpret one layer. The path is shared so those contributions accumulate rather than restart at every handoff.

Evidence changes the route

A gate is useful only when its answer can change the next step. If every answer produces the same path, the question is decorative. Guide authors should make branch consequences explicit and operators should be able to explain why the current route follows from the evidence already committed.

Exit is not silence

An alert disappearing or a user stopping reports is not enough. Exit proof should match the original impact, show the affected and healthy scopes, confirm expected behavior across an appropriate window, and record any residual risk or follow-up owner.

CHAPTER 05 / START HERE

Incident operating rhythm

Coordinate evidence, owners, checkpoints, and communication without turning the tool into another status dashboard.

ROLE PATHS	KEYWORDS	UPDATED
incident-program-leader / technical-account-manager / platform-sre-operations	**incident rhythm**	**2026-07-24**

Establish the clock

At the start of an incident, record the impact window, the next evidence checkpoint, the next decision checkpoint, and any external communication commitment. A checkpoint should have an owner and a concrete expected output. "Update in thirty minutes" is weaker than "Storage owner returns affected-versus-healthy latency evidence by 14:30."

Keep four records distinct

Record	Purpose
Evidence ledger	What was observed, where, when, and by whom
Decision log	What the team decided and the evidence used
Action record	What changed, who authorized it, and expected outcome
Communication record	What was shared, with whom, and the next commitment

Run the narrowing loop

At each checkpoint, ask what changed in the evidence state. Remove boundaries that are no longer supported. Promote boundaries only when evidence supports them. Assign the next unanswered question to a named owner. Avoid reopening settled questions unless new evidence contradicts the previous conclusion.

Close responsibly

Closure requires impact recovery, technical validation, owner agreement, and residual-risk recording. If the immediate service is restored but the underlying boundary remains uncertain, close the active incident only with a separate problem investigation, owner, and due date.

Technical Account Manager playbook

Translate customer pain into a reliable operating record, coordinate owners, and improve escalation quality.

ROLE PATHS	KEYWORDS	UPDATED
technical-account-manager	tam	2026-07-24

Primary responsibility

The Technical Account Manager owns continuity across the customer, technical teams, vendors, and internal decision makers. The role does not require pretending to be the deepest administrator in every domain. It requires making impact, evidence, ownership, commitments, and unresolved risk visible.

What to capture

- Customer-visible impact and affected business process.
- Start time, scope, severity, and change correlation.
- What has been proven versus what is still suspected.
- Named owners for the next evidence and decision gates.
- The next customer update and what it will contain.

What a strong update sounds like

A strong update separates facts from interpretation: "Jobs in queue A are failing after allocation while queue B remains healthy. Scheduler policy and node health are confirmed normal. The storage team owns the next affected-versus-healthy mount comparison by 15:00. No production change is authorized yet."

Expected outputs

The TAM should be able to produce an impact statement, checkpoint plan, owner map, customer-safe update, escalation packet, and final recovery summary. The dossier should reduce the work required for the next team to understand the incident.

Avoid these failure modes

Do not forward raw logs without interpretation, promise a root cause before the boundary is supported, let every team request the same evidence again, or close the customer communication loop before technical validation is complete.

Solutions Engineer and Architect playbook

Turn architecture assumptions into testable boundaries, healthy comparisons, and measurable validation.

ROLE PATHS	KEYWORDS	UPDATED
solutions-engineer-architect	solutions engineer	2026-07-24

Primary responsibility

Solutions Engineers and architects connect the symptom to the system design. Their contribution is strongest when architecture becomes testable: which layer should behave differently if a hypothesis is true, which healthy path provides a fair comparison, and what evidence would move the boundary.

Build the current-state map

Map the affected workload, dependencies, control planes, data paths, identity boundaries, shared services, and operational owners. Keep the map at the level required for the incident. A complete enterprise diagram is not useful if the failure sits in one request path.

Design evidence, not demonstrations

A demo proves that something can work under selected conditions. Incident evidence must explain the affected condition. Define the comparison population, expected signal, access requirement, and interpretation before collecting data. Avoid post-hoc tests that can be interpreted in any direction.

Expected outputs

- Current-state and affected-path map.
- Candidate boundary set.
- Healthy-versus-affected comparison plan.
- Validation design tied to the original impact.
- Architecture constraints and residual risks.

Handoff rule

The architecture owner should make the next owner obvious. A boundary such as "network" is too broad when the evidence supports a specific path, policy, interface, or dependency owner.

Resident and Field Engineer playbook

Capture local proof, preserve command context, and connect physical observations to the shared path.

ROLE PATHS	KEYWORDS	UPDATED
resident-field-engineer	field engineer	2026-07-24

Primary responsibility

The resident or field engineer grounds the incident in local reality. They can verify cabling, power, indicators, environmental conditions, device state, site differences, and command output that remote teams cannot observe directly.

Preserve context

Every command record should include purpose, target, timestamp, access level, expected signals, actual result, and whether the command was read-only or changed state. Screenshots without host, time, or scope often become unusable during escalation.

Compare sites and peers

When possible, compare the affected asset with a known healthy peer using the same method and time window. Record differences in configuration, firmware, topology, load, environment, and recent work. A peer comparison narrows the boundary faster than collecting more unrelated logs.

Stop conditions

Stop and escalate when access is unclear, a command could alter production state, physical safety is uncertain, evidence preservation would be compromised, or the expected target cannot be confirmed. Local presence does not replace authorization.

Expected outputs

Observed state, command evidence, photographs or site notes when appropriate, affected-versus-healthy comparison, local owner confirmation, and a concise handoff to the remote or specialist team.

Platform, SRE, and Operations playbook

Protect production, separate investigation from change, and prove return to service.

ROLE PATHS	KEYWORDS	UPDATED
platform-sre-operations	platform	2026-07-24

Primary responsibility

Platform, SRE, and operations teams protect the production boundary. They decide how evidence may be collected safely, when a controlled change is authorized, what rollback is required, and how service health will be proven after action.

Separate four states

- Referenced command: the guide explains a command and its purpose.
- Externally executed: an authorized operator ran it outside OGO.
- Result captured: evidence from the command was committed to the path.
- Controlled change: an approved state change was executed with rollback and validation.

These states must not be collapsed. Referencing a command is not execution, and execution is not proof of improvement.

Define the safety gate

Before change, confirm scope, authorization, blast radius, rollback, monitoring, stop conditions, and the exact signal expected to improve. Avoid broad changes when a smaller evidence-producing action is available.

Prove return to service

Validation should match the original impact and include system health, workload behavior, user experience, and an observation window appropriate to the failure mode. Record partial recovery honestly. If one path recovers while another remains degraded, the incident state is not fully resolved.

Expected outputs

Change record, health proof, rollback status, return-to-service decision, residual risk, and follow-up owner.

CHAPTER 10 / ROLES

Domain Specialist playbook

Interpret layer-specific evidence and improve the guide without taking over unrelated ownership.

ROLE PATHS	KEYWORDS	UPDATED
domain-specialist	specialist	2026-07-24

Primary responsibility

A domain specialist contributes depth at the point where the evidence reaches their layer. The goal is not to restart the investigation in specialist language. The goal is to interpret the committed evidence, identify the next discriminating signal, and make the supported boundary understandable to the rest of the team.

Enter with context

Review the pain statement, affected scope, healthy comparison, evidence ledger, eliminated boundaries, and current gate before requesting new data. Ask for the smallest additional evidence set that can change the decision path.

Explain confidence

State why the evidence supports or weakens a boundary, what alternative remains, and what would falsify the current interpretation. Avoid absolute claims when the data only supports a probable layer or component class.

Improve the system

After the incident, identify missing guide questions, ambiguous options, weak expected signals, unsafe command language, or ownership gaps. Submit a guide improvement that is reusable and vendor-neutral.

Expected outputs

Technical interpretation, supported fault boundary, confidence statement, next evidence request, owner route, and guide feedback.

Incident and Program Leader playbook

Keep owners, decision clocks, dependencies, and external commitments aligned.

ROLE PATHS	KEYWORDS	UPDATED
incident-program-leader	incident leader	2026-07-24

Primary responsibility

The incident or program leader manages the operating system around the technical path. They make sure every important question, action, approval, communication, and deadline has an owner. They do not replace the technical decision maker.

Make the unresolved work visible

Use the tunnel and dossier to identify unowned branches, overdue evidence, blocked approvals, and cross-team dependencies. A status such as "engineering investigating" is not sufficient. Name the evidence being produced and the next decision it enables.

Time-box intelligently

Time-box a checkpoint according to the cost of waiting and the time required to produce reliable evidence. Short checkpoints that only repeat "still investigating" create noise. Long checkpoints without intermediate proof create risk.

Control communication

Align internal and external statements with the same evidence state. Preserve uncertainty where uncertainty exists. Record the next commitment and who approves the message. Avoid parallel narratives that cause the customer, vendor, and engineering teams to work from different assumptions.

Expected outputs

Checkpoint plan, dependency map, decision log, communication cadence, escalation status, and closure readiness summary.

Engineering and Operations Leader playbook

Make authorization, risk posture, and resource decisions from a visible evidence state.

ROLE PATHS	KEYWORDS	UPDATED
engineering-operations-leader	engineering leader	2026-07-24

Primary responsibility

Engineering and operations leaders decide how much risk the organization will accept, which resources are assigned, whether a change is authorized, and when an incident may move from active response to follow-up work.

Read the evidence state

A leadership readout should answer: what is proven, what remains uncertain, what impact continues, where the candidate boundary sits, who owns the next move, what approval is required, and what happens if the team waits.

Authorize measurable actions

An approval should name scope, intended outcome, rollback, stop conditions, and validation. Avoid approving broad remediation based on urgency alone. When evidence is incomplete, authorize the smallest reversible action that can reduce impact or produce a decisive signal.

Invest after the incident

Use recurring evidence gaps, ownership delays, unsafe manual work, and validation weaknesses to prioritize platform improvements. The goal is not merely fewer incidents; it is faster boundary narrowing and safer recovery when incidents occur.

Expected outputs

Risk posture, authorization decision, resource assignment, operating readout, residual-risk acceptance, and investment follow-up.

Technology domain walkthroughs

Apply the same operating logic across HPC, AI platforms, cloud, networking, storage, and data-center operations.

ROLE PATHS	KEYWORDS	UPDATED
everyone / domain-specialist	hpc	2026-07-24

HPC and AI infrastructure

GPU, fabric, scheduler, thermal, memory, storage, and workload symptoms cross ownership boundaries. Define the affected job population, compare healthy nodes or queues, and separate workload behavior from shared platform behavior before assigning a hardware or scheduler boundary.

AI platform

A model-serving symptom may arise from model quality, capacity, dependency latency, policy, data, orchestration, or request shaping. Preserve request class and healthy-path comparisons. Do not label a quality issue as infrastructure without evidence, or an infrastructure issue as model behavior because the output is visible at the application layer.

Cloud

Cloud symptoms can sit in identity, quota, network, service control plane, workload configuration, regional dependency, or provider service. Use region, account, request class, and dependency comparisons to avoid treating the first visible service as the fault boundary.

Networking

Define source, destination, path, protocol, time window, and expected behavior. Separate reachability, loss, latency, policy, name resolution, and application response. A packet path should be paired with control-plane and endpoint evidence when relevant.

Storage

Separate capacity, latency, throughput, pathing, metadata, permissions, and application access patterns. Compare the affected mount, volume, or object path with a healthy peer under similar load.

Data center and physical systems

Power, cooling, rack position, cabling, environmental conditions, firmware, and maintenance history matter. Physical work requires explicit safety and authorization. Preserve photographs, labels, timestamps, and the exact asset identity.

Evidence quality and command safety

Collect decision-changing evidence while preserving authorization, context, and production safety.

ROLE PATHS	KEYWORDS	UPDATED
everyone / platform-sre-operations / resident-field-engineer	evidence quality	2026-07-24

Evidence quality standard

Evidence is useful when it is attributable, scoped, timed, reproducible enough for review, and connected to a decision. A copied error string without source and time is a clue, not a complete evidence record.

Command record requirements

Field	Required meaning
Purpose	Which question the command answers
Target	Exact system, scope, or object
Access	Required privilege and owner
Safety	Read-only, controlled change, or prohibited
Expected signals	Outcomes and how they affect the path
Actual result	Captured output or observation
Timestamp	When the state was observed

Investigation versus change

Read-only collection can still create load, expose sensitive data, or require elevated access. Controlled change requires authorization, rollback, monitoring, and validation. OGO should explain the boundary; it should not imply that a command is safe simply because it appears in a guide.

Stop conditions

Stop when the target is uncertain, the command differs from the approved form, the result indicates a larger blast radius, rollback is unavailable, access is not authorized, or evidence preservation could be lost. Record the stop and route it to the accountable owner.

Redaction and handling

Remove secrets, tokens, personal data, customer identifiers, and unnecessary payload content before sharing. Preserve the structure required for interpretation without copying the entire environment into the dossier.

Optional AI, BYOK, and privacy

Understand deterministic operation, per-session keys, hosted trust boundaries, and prohibited logging.

ROLE PATHS	KEYWORDS	UPDATED
everyone / engineering-operations-leader	ai	2026-07-24

Deterministic first

The core simulator and approved guide path operate without AI. Optional AI can help classify a pain statement or summarize a record, but its output is advisory. Evidence committed by the operator remains the source of truth.

Per-session BYOK

A user-provided API key should remain isolated to the current browser session or request path according to the deployed implementation. It must not be written to local storage, analytics, logs, source control, or a public client environment variable. Never prefix a secret with NEXT_PUBLIC_.

Hosted environment key

A server-side OPENAI_API_KEY belongs in the hosting provider's secret settings. Anonymous public traffic should not inherit an unrestricted owner key by default. Use explicit policy, rate limits, usage caps, monitoring, and a server-only boundary.

Safe logs

Safe logs may include request ID, deployment mode, region, model name, latency, token counts, success or failure state, and redacted error category. They must not include API keys, full prompts containing customer data, raw evidence payloads, access tokens, or secrets returned by a provider.

Multi-user boundary

Each request must be associated with its own session context. User A must never receive User B's key, prompt, evidence, or response. Hosted relays should avoid shared mutable key state and should redact provider errors before returning them to the browser.

Dossiers and escalation packets

Produce a stable evidence package that another team can understand without reconstructing the incident.

ROLE PATHS	KEYWORDS	UPDATED
technical-account-manager / incident-program-leader / domain-specialist	dossier	2026-07-24

Purpose of the dossier

The dossier is a decision-ready incident record, not a dump of every message and log. It should let a qualified reviewer understand impact, scope, evidence, supported boundaries, actions, ownership, and validation without calling the original operator.

Required sections

- Incident identity, time window, and customer or service impact.
- Affected scope and healthy comparison.
- Evidence ledger with timestamps and sources.
- Candidate boundaries, eliminated boundaries, and current confidence.
- Command and action record with authorization.
- Owner handoff and next checkpoint.
- Escalation ask and the evidence already collected.
- Exit criteria, recovery proof, and residual risk.

Escalation quality

A strong escalation asks for a specific interpretation, action, or approval. It shows why the receiving team is the correct owner and prevents repeated evidence requests. "Please investigate" is weak. "Confirm whether the affected-versus-healthy controller state supports a control-plane boundary and identify the safe next collection step" is actionable.

Publication safety

Remove secrets and unnecessary identifiers. Mark assumptions as assumptions. Do not claim root cause when the dossier supports only a boundary. Preserve enough technical detail for review while respecting customer and security policy.

Contributing diagnostic guides

Turn repeatable operating pain into a safe, vendor-neutral, reviewable guide.

ROLE PATHS	KEYWORDS	UPDATED
contributor / domain-specialist	contribute	2026-07-24

Start with a repeatable pain

A guide should address one recognizable operating problem with a stable evidence path. Avoid a universal guide for an entire technology domain. The pain statement should include common operator language, alerts, and symptom examples without embedding customer data.

Build the evidence graph

Each question should define the evidence being requested, answer options, and next target. Branches must resolve to another step or a complete resolution. Every answer should have a meaningful consequence; decorative questions increase operator effort without improving the boundary.

Add ownership and safety

Identify who can provide the evidence, who authorizes change, who performs the action, who receives escalation, and who validates recovery. Commands must include purpose, expected signals, access, and safety classification.

Review before submission

Run schema, graph, domain-isolation, language, TypeScript, and test validation. Read the guide as an operator who did not author it. Confirm that the current state, next owner, safety gate, and exit proof are always understandable.

Contribution outcome

A merged guide should improve the shared public knowledge base without exposing private platform logic, customer information, proprietary vendor procedures, or unsafe automation.

JSON guide authoring reference

Use stable identifiers, explicit branches, command metadata, owner routes, and measurable resolutions.

ROLE PATHS	KEYWORDS	UPDATED
contributor / domain-specialist	json	2026-07-24

Stable anatomy

A guide normally includes identity, title, description, technology domain, diagnostic area, operator phrases, scope guidance, steps, commands, owner routes, and resolution definitions. Follow the repository template and existing schema rather than inventing parallel fields.

Step design

Each step needs a stable ID, a clear evidence question, and options whose next target exists. Questions should request observable evidence. Avoid compound questions that can produce two conflicting answers.

```
{
  "id": "affected-peer-comparison",
  "question": "Does the healthy peer show the same controller state?",
  "options": [
    { "id": "same", "label": "Same state", "next": "shared-boundary-check" },
    { "id": "different", "label": "Different state", "next": "local-controller-check" },
    { "id": "unknown", "label": "Not yet known", "next": "collect-peer-state" }
  ]
}
```

Resolution design

A resolution should include diagnosis, supported boundary, confidence, nextAction, safety, owner route, escalation packet, and measurable validation. Use nextAction; do not introduce an incompatible action property.

Command metadata

Document purpose, command text or reference, expected signals, required access, safety classification, stop conditions, and whether execution occurs outside OGO. Never embed secrets or destructive defaults.

Validation, testing, and review quality

Require schema, graph, safety, language, runtime, and user-experience quality together.

ROLE PATHS	KEYWORDS	UPDATED
contributor / engineering-operations-leader	**validation**	**2026-07-24**

Required command set

```
npm run validate:docs
npm run validate:playbooks
npm run typecheck
npm run test
npm run dev
```

The documentation validator checks frontmatter, categories, role aliases, duplicate slugs, required content, and publication links. The playbook validator checks JSON structure, graph targets, domain isolation, question counts, and protected contracts.

Reviewer responsibilities

Reviewer	Primary focus
Domain reviewer	Technical accuracy, evidence logic, expected signals
Safety reviewer	Access, controlled change, stop conditions, preservation
Documentation reviewer	Clarity, consistency, vendor-neutral language
QA reviewer	Schema, graph, tests, routes, responsive behavior
Maintainer	Project fit, release quality, attribution, merge decision

Knowledge-base test

A qualified operator who did not write the guide should be able to explain the current evidence state, next owner, safety gate, and validation without contacting the author.

Regression policy

Tests should protect behaviors and contracts, not obsolete marketing copy. A wording update should not break the suite when the same intake, isolation, or safety behavior remains present. Exact strings are appropriate only for protected names, API fields, or safety statements that must not drift.

Adoption and governance

Introduce OGO through controlled practice, review ownership, release discipline, and measurable operating outcomes.

ROLE PATHS	KEYWORDS	UPDATED
engineering-operations-leader / incident-program-leader / contributor	**adoption**	**2026-07-24**

Start with a bounded pilot

Choose a small set of repeatable incidents and a cross-functional group that already works together. Run tabletop scenarios before production use. Measure whether teams reach a clearer boundary, reduce duplicate evidence requests, and produce stronger handoffs.

Define ownership

Assign maintainers for guide review, safety review, release approval, and publication updates. Define who can merge guides, who can change protected contracts, and how deprecated guides are handled.

Version the knowledge base

Treat guide and documentation changes as product releases. Record version, date, changed behavior, migration notes, and validation status. The hosted documentation should update with the repository, while downloadable publications show their edition clearly.

Measure useful outcomes

Useful adoption measures include time to first bounded hypothesis, repeated evidence requests avoided, unowned checkpoints, escalation acceptance, controlled-change completeness, return-to-service proof quality, and guide improvements submitted after incidents.

Avoid vanity adoption

Page views and guide counts do not prove operating value. A large catalog with weak paths creates false confidence. Prefer fewer guides with clear evidence, ownership, safety, and validation.

Workshop and tabletop facilitation

Run a structured team exercise that teaches evidence discipline, owner routing, safety, and closure.

ROLE PATHS	KEYWORDS	UPDATED
incident-program-leader / technical-account-manager / contributor	workshop	2026-07-24

Recommended 60-minute format

13. Five minutes: explain the product boundary and scenario.
14. Ten minutes: capture reported pain, impact, scope, and prohibited outcomes.
15. Twenty minutes: run evidence gates and use the expanded tunnel.
16. Ten minutes: assign owners, authorize or reject a controlled move, and define validation.
17. Ten minutes: produce a dossier and customer-safe update.
18. Five minutes: debrief the operating behavior.

Participant roles

Assign an operator, customer or service lead, technical owner, safety or change approver, specialist, and observer. The observer tracks guessing, duplicate evidence, unowned work, unclear commands, and premature closure.

Facilitator prompts

Ask: What changed in the evidence state? Which boundary was eliminated? Who owns the next answer? What must be true before change? What proves recovery? Do not rescue the team with the answer; keep them inside the operating process.

Debrief

Review where the team confused observation with conclusion, where role boundaries helped or hindered, whether the dossier could stand alone, and which guide wording should improve. Capture changes as repository issues or pull requests.

CHAPTER 22 / START HERE

FAQ and glossary

Resolve common questions and align the language used across guides, incidents, and publications.

ROLE PATHS	KEYWORDS	UPDATED
everyone	faq	2026-07-24

Frequently asked questions

Does OGO run commands?

The guide can reference and explain approved commands. Execution remains external and human-authorized unless a future explicitly controlled integration says otherwise. The record distinguishes reference, execution, captured result, and controlled change.

Is AI required?

No. The deterministic simulator and local guide path work without AI. Optional AI output does not become proof unless an operator verifies and commits supporting evidence.

Is a fault boundary the same as root cause?

No. A boundary is the smallest supported area where the evidence places the problem. Root cause may require additional engineering, reproduction, or vendor analysis.

Why keep PDF and DOCX files?

They provide fixed offline editions for workshops, review, records, and users who cannot access the hosted site. The hosted documentation remains the current searchable source.

Glossary

Evidence gate: A question whose supported answer changes the route. Committed evidence: An observation the operator has accepted into the incident path. Candidate boundary: A technical or ownership area still supported by current evidence. Owner route: The person or team responsible for the next evidence, decision, action, or validation. Safety gate: Conditions that must be satisfied before a controlled action. Exit proof: Measurable evidence that the original impact has recovered. Dossier: A stable decision-ready incident and escalation record. Healthy peer: A comparable unaffected path, asset, site, service, or workload used to narrow differences.

Offline publication library

Download versioned field editions and understand which operating need each publication serves.

ROLE PATHS	KEYWORDS	UPDATED
everyone / contributor	**pdf**	**2026-07-24**

Current publication

The OGO User and Team Operating Playbook Documentation Edition is available as PDF and editable DOCX. It mirrors the hosted chapter structure and carries an edition date. Use it for offline reading, facilitated workshops, formal review, and controlled distribution.

- Download the PDF
- Download the editable DOCX

Recommended future publications

Quick Start Field Manual: Installation, first incident, evidence gates, tunnel use, dossier export, and common mistakes in approximately ten pages.

Role Field Cards: One or two pages per role showing responsibilities, evidence owned, decisions authorized, handoff expectations, and required outputs.

Security, Privacy, and BYOK Trust Brief: Local-first behavior, optional AI boundaries, per-session key handling, hosted policy, safe logs, and prohibited logs.

Incident Scenario Atlas: Visual scenarios for HPC, AI infrastructure, cloud, networking, storage, and data-center operations.

Guide Authoring and Validation Handbook: JSON anatomy, branch logic, command metadata, safety labels, owner routes, review gates, and release checks.

Workshop Facilitator Kit: Agenda, roles, prompts, observation sheet, debrief questions, and scoring rubric.

Publication rule

A publication is a versioned snapshot, not a second knowledge base. Corrections should land in repository Markdown and the hosted site first, then appear in the next offline edition.

One-page operating reference

1. REPORTED PAIN	Impact, scope, time, correlated change, existing proof, prohibited outcome.
2. COMMITTED EVIDENCE	Observable, attributable, scoped, timed, and connected to a decision.
3. CURRENT GATE	One question whose answer changes the route.
4. BOUNDARY	Supported area, eliminated alternatives, confidence, and remaining uncertainty.
5. OWNER ROUTE	Named owner for evidence, decision, action, escalation, and validation.
6. SAFETY GATE	Authorization, target, blast radius, rollback, monitoring, stop conditions.
7. EXIT PROOF	Original impact recovered, healthy comparison, observation window, residual risk.

THE CLOSURE QUESTION

Could a qualified operator who did not participate explain the current evidence state, next owner, safety gate, and validation from the record alone?