DevBridge Subscription Enforcement PRD

v0.1 – Quota, Grace Period, and Safe Degradation Rules

Document owner	Product / Platform
Audience	Engineering, QA, Product Ops, Support, Finance
Status	Draft for implementation
Last updated	2025-12-27
Scope	Plan entitlements + quota enforcement across SDK + backend + UI

1. Summary

DevBridge provides a mobile-first platform (SDK + dashboard) for API traces, screen flow timelines, device logs/crashes, remote configuration, localization, monitoring, and cost analytics. Subscription enforcement must drive upgrades without breaking customer apps. The system must enforce plan quotas via safe degradation (sampling, retention reduction, edit freezes) rather than disabling the SDK.

2. Goals and Non-goals

2.1 Goals

- Enforce Free/Pro/Team/Enterprise quotas consistently across all modules.
- Prevent 'SDK disabled' scenarios that could disrupt production apps.
- Provide clear upgrade triggers: usage visibility, warnings, grace period, and predictable degradation.
- Support monthly renewal with usage reset and accurate metering/aggregation.
- Enable enterprise entitlements (SSO/SAML, SCIM, custom retention, data controls) without affecting SMB flows.

2.2 Non-goals (v0.1)

- Complex proration and mid-cycle plan changes beyond basic upgrade handling.
- Invoicing, tax, and payment processor details (covered in Billing PRD).
- Advanced anomaly detection or AI recommendations (future).
- Multi-region residency routing (future enterprise add-on).

3. Guiding Principles

- Never break the customer app: the SDK continues to run even when over quota.
- Last-published content remains available: remote config and localization keep serving the latest published snapshot.
- Prefer 'degrade gracefully' over hard stops: sampling and retention are the primary levers.

- Make limits understandable: show the exact metric, current usage, reset date, and what changes at each threshold.
- Security by default: request/response bodies are captured only when explicitly enabled and allowed by plan/policy.

4. Personas and Upgrade Triggers

Persona	Primary jobs	What causes upgrade
Solo developer / learner	Debug crashes and API issues	Needs longer retention or higher session/request limits
QA / SDET	Reproduce issues and validate fixes	Needs more projects, exports, build comparison, mocking
Startup team	Ship faster with shared visibility	Needs seats/RBAC, environments, integrations, higher quotas
Ops / Reliability	Alerting and release health	Needs advanced alerts, on-call routing, custom retention
Enterprise	Governance and compliance	SSO/SCIM, audit logs, data controls, custom limits, support SLA

5. Plan Structure and Metering Dimensions

Plans: Free, Pro, Team, Enterprise. Each plan defines entitlements and quota limits. Enterprise can override limits per metric.

5.1 Common quota dimensions

- Monthly active devices (MAD): distinct device_id reporting within the billing month
- Sessions: distinct session_id started within the billing month
- API requests/traces: count of captured network events (after sampling)
- Ingest volume: total bytes accepted by ingestion services (compressed payload size)
- Stored data volume: bytes retained after compaction
- Retention: days of searchable data per module
- Projects/apps: number of applications/environments per organization
- Seats: active users with access to the org
- Alert rules: number of active alert rules and notification targets
- Config/Localization: number of keys, and publish operations per month

5.2 Metering sources of truth

- Server-side ingestion counters are the source of truth for billable usage.
- SDK-side counters are advisory (used for local UX and pre-flight warnings), not billing.
- Usage is aggregated daily and monthly by org_id, app_id, env, and module.

6. Enforcement Model

6.1 States

State	Trigger	User impact	Goal
ACTIVE	<80% quota	Normal operation	Adoption
WARN	>=80% and <100%	Banners + email warnings; no behavior change	Prevent surprise
GRACE	>=100% (grace window)	Continue ingest at full fidelity for 24-72h (plan-configurable)	Convert without disruption
DEGRADED	Grace expired and still >=100%	Sampling + reduced retention + feature freezes	Force upgrade trigger safely
SUSPENDED (rare)	Abuse, unpaid invoice, or explicit admin action	Ingestion blocked; serving last-published config/locales continues	Protect platform

6.2 Thresholds (defaults)

Thresholds are configurable per plan and per metric. Default policy:

Threshold	Default	Behavior
Warn threshold	80%	In-product banner + email
Hard threshold	100%	Enter GRACE
Grace period	48 hours	No degradation during grace
Overage buffer	Up to 10%	Optional buffer before DEGRADE for specific metrics

6.3 What we never do

- Do not disable the SDK or crash the app.
- Do not stop serving last-published config/localization values.
- Do not silently drop data without reporting it in UI (must show 'sampled' / 'dropped').

7. Module-by-Module Enforcement Rules

API Traces

Rule	Behavior
Counts	Captured HTTP(S) request/response events after sampling; bytes ingested
Degrade	Switch to sampling (e.g., 1:10 sessions or 1:20 requests) and/or drop response bodies first
Hard stop	Block ingestion only in SUSPENDED or extreme abuse
UI signal	Mark charts and tables with 'Sampled' and show effective sampling rate

Screen Flow & Timeline

Rule	Behavior
Counts	Sessions and timeline events per session; bytes ingested
Degrade	Sample sessions first (keep all events for sampled-in sessions); cap per-session event count

Retention	Reduce retention days for Free/over-limit orgs
UI signal	Show 'Partial sessions captured' indicator

Device Logs & Crashes

Rule	Behavior
Counts	Log events + crash events; bytes ingested; retained volume
Degrade	Prioritize crashes/ANR over verbose logs; drop DEBUG logs first
Retention	Shorten retention; keep crash group summaries longer where possible
UI signal	Explain which log levels are captured

Monitoring & Alerts

Rule	Behavior
Counts	Alert rules, notification targets, and evaluation events
Degrade	Disable advanced alert types; keep basic health alerting for paid plans
Hard limit	Prevent creation of new rules; keep existing rules running until SUSPENDED
UI signal	Show 'Rule creation locked (over limit)'

Remote Business Config

Rule	Behavior
Counts	Keys, environments, and publish operations; request volume (config fetches)
Degrade	Freeze editing/publishing when over limit; continue serving last published snapshot
Safety	SDK always returns cached last-known-good and uses backoff
UI signal	Banner: 'Publishing disabled until upgrade'

Localization

Rule	Behavior
Counts	Keys, locales, and publish operations; SDK fetch volume
Degrade	Freeze publishing; serve last published bundle; optionally limit new locales
Retention	Not applicable (content store) - enforce by key/locales/publishes
UI signal	Show key count and publish cap

Cost Analytics / API Monetization

Rule	Behavior
Counts	Derived from traces and session volume; dashboards compute-only
Degrade	Restrict advanced breakdowns/exports; keep basic totals visible to encourage upgrade

Hard limit	No ingestion changes beyond traces; this is a presentation entitlement
UI signal	Lock icon on premium reports

Build Versioning

Rule	Behavior
Counts	Build uploads and retained build history
Degrade	Cap number of builds retained; older build metadata archived
Hard limit	Prevent new build registration if build quota exceeded (still allow SDK to report build id)
UI signal	Show build history cap

8. User Experience Requirements

8.1 Admin usage dashboard

- Show usage meters per dimension with: current usage, quota, percent, projected end-of-month, reset date.
- Show enforcement state (ACTIVE/WARN/GRACE/DEGRADED) and effective sampling rate per module.
- Provide 1-click upgrade CTA; show estimated plan required based on current run-rate.

8.2 In-product banners and locking

When	Where	Message	Action
WARN (>=80%)	Dashboard + email	You are approaching your monthly quota for <metric>.	View usage / Upgrade
GRACE start	Dashboard + email	Quota exceeded. Grace period ends on <timestamp>.	Upgrade / View impact
DEGRADED	Dashboard + module pages	Data is sampled or retention reduced due to quota limits.	Upgrade / See details
Publish frozen	Config/Localization editors	Publishing is disabled until you upgrade or quotas reset.	Upgrade

8.3 Upgrade education (what changes)

On every limit breach, show a 'What changes if you upgrade' panel with the next plan’s exact new limits and features.

Minimum required content:

- Limits increased (devices, sessions, requests, retention)
- Team features (seats, RBAC, environments)
- Ops features (alerts, integrations)
- Governance (audit logs, SSO) for Enterprise

9. SDK Behavior

9.1 Enforcement configuration fetch

- SDK fetches entitlements/enforcement config at app start and every N minutes with exponential backoff (default: 15 min).
- SDK caches last-known-good entitlement response for 24h; if offline or service unavailable, continue with cached config.
- Enforcement config includes: sampling rates, allowed payload fields (e.g., body capture), max event buffer size, and publish permissions (for admin-only SDK flows if any).

9.2 Safe degradation in SDK

Over-limit condition	SDK behavior
Sampling enabled	Apply sampling decision at session start; all events in sampled-in sessions are captured
Body capture disabled	Do not collect request/response bodies; keep headers/metadata based on policy
Buffer cap reached	Drop oldest low-priority events first; always keep crashes/errors
Config/localization publish frozen	No SDK impact on reading; admin publish calls return 'frozen' status

10. Backend Architecture

10.1 Services

- Entitlement Service: resolves plan -> features/limits; supports enterprise overrides
- Usage Metering Service: accepts ingestion counters and aggregates by day/month
- Quota Evaluator: computes threshold states and produces enforcement policy outputs
- Enforcement Policy Service: publishes effective policy to SDKs and dashboard
- Notification Service: email + in-app + webhook events; rate-limited and deduplicated

10.2 Data model (high-level)

Entity	Fields (examples)
Organization	org_id, name, billing_account_id, timezone
Subscription	subscription_id, org_id, plan_id, status, period_start, period_end
Plan	plan_id, name, limits_json, features_json
EnterpriseOverride	org_id, metric, limit_value, retention_days
UsageAggregateMonthly	org_id, metric, value, period_start, updated_at
EnforcementState	org_id, state, grace_end_at, effective_policy_json
NoticeLog	org_id, type, sent_at, dedupe_key

10.3 APIs (v0.1)

API	Purpose	Auth
GET /api/entitlements	Return plan features/limits for current org	User token
GET /api/usage/current	Return current period usage + projections	User token

GET /api/enforcement/policy	Return effective policy for SDK (sampling, capture rules)	SDK key
POST /api/usage/event	Optional: post client advisory counters (non-billing)	SDK key
POST /api/admin/config/publish	Publish config/locales; enforce freeze	User token

11. Edge Cases

- Upgrade mid-cycle: new limits apply immediately; state transitions re-evaluated within 5 minutes.
- Downgrade mid-cycle: apply at next renewal by default (avoid sudden degradation).
- Delayed ingestion: late events count toward the period they are received (v0.1); document this clearly.
- Multiple apps/environments: quotas count at org level by default; enterprise may allocate per app.
- Abuse protection: rate limits independent of plan quotas; SUSPENDED is reserved for abuse/unpaid.

12. Security and Compliance

- All enforcement policies served to SDK must be signed (HMAC) to prevent tampering.
- Role-based access: only admins can change capture settings and publish config/locales.
- Audit log: record changes to capture settings, quotas (enterprise overrides), and enforcement state overrides.
- PII/Secrets: default redaction; advanced redaction rules are paid/enterprise feature.

13. Metrics and Analytics

Metric	Definition
Over-limit rate	% org-months that enter GRACE
Conversion from GRACE	% orgs upgrading within grace window
Degraded duration	Median time org stays in DEGRADED before upgrade/reset
Churn after limit	% orgs uninstall/stop sending after first over-limit event
Top quota drivers	Which metric most frequently triggers upgrades

14. Rollout Plan

- Phase 0: internal dogfood with debug UI for enforcement decisions.
- Phase 1: beta for new orgs only (Free/Pro) with conservative grace period and clear messaging.
- Phase 2: enable for all orgs; start enforcing publish-freeze for config/localization first (low risk).
- Phase 3: enable sampling-based degradation for traces/flows/logs after monitoring error rates.

15. Acceptance Criteria

- Over-limit orgs never experience SDK crash or app disruption caused by DevBridge.
- At >=80% usage, users see a banner and receive at least one email warning per period (deduped).
- At >=100% usage, org enters GRACE with a recorded grace_end_at and visible countdown in UI.
- After grace_end_at, enforcement switches to DEGRADED and effective sampling/retention changes take effect within 15 minutes.
- Publishing for config/localization is blocked in DEGRADED but serving last-published remains functional.

16. Test Plan

- Unit tests: quota evaluator thresholds, state transitions, enterprise overrides precedence.
- Integration tests: ingestion -> usage aggregation -> enforcement policy update -> SDK fetch behavior.
- Load tests: usage aggregation at scale; verify evaluator runs within SLA.
- Chaos tests: entitlement service unavailable; SDK uses cache; no app impact.
- Security tests: policy signature verification; attempt bypass by replaying old policies.

17. Open Questions

- Grace duration by plan: 24h (Free) vs 72h (Team)?
- Default sampling strategy: session-based vs request-based per module?
- Should Free allow a minimal number of alerts, or alerts only in paid?
- How to handle late-arriving events (received after period_end)?