Engineer

A bounded event pipeline for a macOS app

Bounded event buffers and logging after a recorded macOS incident with 41 MB per second growth and 111 GB of compressed pages on a 36 GB machine; a recorded full test run reduced core log lines from 610,996 to 1,411.

September 2026

How events reach consumers and logs

Implementation overview of event delivery and logging; the case distinguishes full test-run log volume from user-session performance.

1.

Native events

Native code publishes application events for consumers.

2.

Typed, bounded subscriptions

Typed, bounded subscriptions deliver events and retain a limited history.

3.

Log budget

The log budget controls how much event detail reaches written logs.

41 MB

recorded memory growth per second behind a buffer redesign

Situation. On 2026-09-01, diagnostic reports recorded 111 GB of compressed pages for the macOS app on a 36 GB machine. A separate footprint sample recorded growth of about 41 MB per second while a connection retry loop emitted core events.

Task. The event path needed explicit memory bounds. A burst of core log events was retained in an unbounded main stream and copied to seven subscriber streams whose consumers did not use those log events.

Action. Buffers were bounded, subscriptions filtered by event type and logging given a rate budget. Overflow handling makes the trade explicit: repetitive diagnostic events may be dropped, and affected state can be refreshed rather than retaining unlimited events. Architecture checks and application-budget tests defend the bounded design.

Result. The incident record documents a full test-run comparison: core log lines fell from 610,996 to 1,411, and captured log size from 63 MB to 388 KB. These are recorded test-run measurements, not a production traffic benchmark or a measured post-fix memory figure. The logging change alone did not fix separate shutdown timeouts; those required their own correction.

Bring this experience to your project

Use “A bounded event pipeline for a macOS app” as a starting point for a similar challenge in your organisation. The services behind this work are available for your project. The shape of a system worked out on paper first, so the parts fit and the cost is known before anyone writes code.

Explore our services Get in touch →

Look behind the result

Read the situation, decisions and result in this example. We can discuss which parts transfer to your constraints and which need another approach. A bounded event pipeline for a macOS app — 41 MB recorded memory growth per second behind a buffer redesign

Explore our services Get in touch →

Is your app quietly eating the machine's memory?

Talk through your project with the engineer responsible for delivery. We agree the scope, price basis and handover in writing before work begins.

Role context: macOS Application Engineer. It draws on Solution Architecture, Frontend Engineering, Reliability & Backups, Monitoring & Observability, Performance Tuning and Testing & QA. And while you are here, we are rather good at Platform & Solution Architecture, Frontend Development and Site Reliability & Monitoring.

Get in touch Explore our services

One piece of **a longer record** — and it is all on this site.

Every entry is written the same way: the situation, the task, what we did and what changed.

See the whole record

Define the improvement you need

Describe the workflow, the existing interfaces and the users who need the change. We can define a testable delivery and a handover that lets your team maintain it.

See what it costs Get in touch →

Is your app quietly eating the machine's memory?

Ask about the decisions behind “A bounded event pipeline for a macOS app”. You will speak with the people who can explain the work and assess your own project.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/portfolio/an-app-that-ate-41-mb-a-second-stopped/