semaphore

How does Semaphore Open Source work?

Opening Up Semaphore: Building the Future of CI/CD in the Open

A Technical Deep Dive into the Architecture, Strategy, and Community-Driven Development of Semaphore

---

Executive Summary

Semaphore, the trusted CI/CD platform used by thousands of engineering teams, is entering a new era by open-sourcing its core technology. With a modular microservices backbone, a robust orchestration engine, and a collaborative open source ethos, Semaphore's Community Edition is purpose-built for teams seeking flexibility, transparency, and control.

This white paper outlines the motivations behind this strategic move, the architectural principles that power Semaphore, how it was opened up, and the future it enables for developers and organizations.

Table of Contents

---

1. Foreword

In 2010, we were a team of five consultants, installing Jenkins and debugging plugin failures. That frustration eventually sparked Semaphore. We launched our hosted CI/CD service in 2012, starting with a strong focus on the Ruby on Rails community. Over time, the product evolved through a full rewrite, the introduction of an enterprise edition, and support for on-premise deployments.

In 2025, we reached a milestone: Semaphore is now open source. In this white paper, we the journey so far in open sourcing our platform.

---

2. Why Go Open Source?

Many teams loved Semaphore's speed and reliability, but couldn't use it due to compliance constraints or infrastructure limitations. By going open source, we're removing those barriers.

Open-sourcing Semaphore lets developers deploy it anywhere — on a laptop, in a private cloud, or behind a firewall — and encourages public collaboration, resulting in a stronger, more inclusive product.

---

3. Developer-Led Design: Building the Community Edition

Cloud platforms present fundamentally different challenges from open source. Cloud platforms provide a built-in structure. This includes technical support, automated maintenance, and predictable infrastructure. Open source solutions, however, offer none of these advantages. This disparity forced us to completely rethink our onboarding approach for the Community Edition.

We focused on:

- Simple installation: Using Helm and Terraform to simplify setup
- Self-guided onboarding: Get started quickly
- Documentation as product: We redesigned the docs with first-time users in mind

We tested this with real users. We recorded their sessions, measured time-to-first-job, and iterated until we could deliver success without any hand-holding

4. Testing in the Open: Ephemeral Environments

Semaphore's internal CI/CD was deeply tied to our infrastructure. But for open source to work, we had to prove Semaphore runs cleanly on customer hardware and any cloud provider.

We built ephemeral test environments to ensure Semaphore works everywhere:

- GCP: We created a minimal Terraform setup with Helm charts, DNS reservations, and static IPs
- AWS: We added EKS-specific tooling like AWS LB Controller and External DNS
- Local testing: we focused on running Semaphore on a single VM, ensuring fast-booting a k3s cluster for dev iteration

Each release spins up fresh environments, runs end-to-end tests, and tears them down. It's automated, portable, and repeatable.

---

5. Under the Hood: Semaphore Architecture and Orchestration

Semaphore runs on a distributed system of over 30 microservices built in Elixir and Go, and orchestrated by Kubernetes.

Core Services

- Plumber: Orchestrates pipelines using a state machine
- Zebra: Assigns jobs to agents
- Log Hub: Streams, stores, and archives logs

Communication Stack

- gRPC for internal messaging
- HTTP for user/API interaction
- RabbitMQ for asynchronous task queues

Agent Model

- Hosted or self-hosted: Agents run jobs either in Semaphore's cloud or inside your infrastructure

- Quota-controlled: We use Postgres advisory locks to prevent rogue agents from overusing resources
- Stateful lifecycle: We more clearly manage transitions through job lifecycle states

6. One API to Rule Them All

Semaphore's early API was fragmented. We rebuilt them into a unified, resource-oriented API inspired by Kubernetes:

```JSON
{
  "kind": "project",
  "metadata": { "id": "proj-123" },
  "spec": {
    "name": "My App",
    "description": "CI/CD pipeline"
  }
}
```

Features:

- Rich object representations (e.g., full user info, not just IDs)
- Resource-scoped custom methods (e.g., POST /projects/{id}/start)
- Typed schemas, validation, and permission checks via OpenAPI-first development

This unified API makes Semaphore easier to reason about, document, and extend, especially in the open.

---

7. Building in Public

Open source isn't just a licensing change; it's a new way of working.

We now:

- Use a public GitHub repository for product and code
- Host discussions and SIGs (special interest groups) on Discord
- Share designs and product reviews on YouTube via Semaphore Backstage
- Run meetings and release planning transparently

We're also creating:

- Clear contribution guidelines
- Beginner-friendly issues
- Dedicated maintainer hours

We aim to design in the open, with the community's help.

---

8. The Road Ahead

We're just getting started. Here's what's next:

- Monthly releases for faster feedback
- Continued API improvements with better errors and discovery tools
- New testing scenarios across clouds
- Contributor roles and SIG expansions

We believe open source should come with great UX, deep testing, and long-term support.

---

9. Use Case: Semaphore in an Air-Gapped Fintech Environment

A mid-sized fintech company with strict compliance requirements wanted to migrate from a legacy CI/CD setup to Semaphore Community Edition. The goal was to perform Kubernetes deployments in an air-gapped network. This meant a fully self-hosted setup with audit logs.

Challenges

- No access to public cloud or external registries
- Compliance rules around access control and audit trails
- Need for repeatable test environments for each deployment

How They Solved It with Semaphore

- Self-hosted agents are deployed via Kubernetes inside their secure network
- Secrets management is integrated with their internal vault system
- Pipeline audit logs are exported to internal SIEM tools via custom webhooks

- Ephemeral test environments spun up using k3s to validate microservice deployments pre-production

Outcomes

- Reduced deployment times by 40%
- Full compliance with internal and regulatory controls
- Ability to run Semaphore entirely offline with air-gap scripts

This case highlights Semaphore's value when flexibility, control, and transparency are critical.

---

10. Appendix: Examples and Resources

Sample Pipeline YAML

```
None
version: v1.0
name: Build and Test
blocks:
  - name: Build
    task:
      jobs:
        - name: Compile
          commands:
            - make build
  - name: Test
    task:
      jobs:
        - name: Unit Tests
          commands:
            - make test
```

Sample API Response

```JSON
{
  "kind": "workflow",
  "metadata": {
    "id": "wf-001",
    "created_at": "2025-06-01T12:00:00Z"
  },
  "spec": {
    "project_id": "proj-456",
    "status": "running"
  },
  "created_by": {
    "kind": "user",
    "id": "user-789",
    "name": "alice"
  }
}
```

Helm Install Snippet

```Shell
helm upgrade --install semaphore oci://ghcr.io/semaphoreio/semaphore \
 --version v1.3.0 \
 --set global.domain.name="example.com" \
 --set ingress.ssl.certName="example-certificate"
```

Useful Links

- GitHub repo: https://github.com/semaphoreio/semaphore
- Public API docs: https://docs.semaphoreci.com/reference/api
- Discord: https://discord.gg/FBuUrV24NH
- Open Roadmap: https://github.com/semaphoreio/semaphore/blob/main/ROADMAP.md

11. Conclusion

We didn’t just flip a switch to go open source. We have to rethink how Semaphore is built, tested, and shared. The result is a robust CI/CD engine that developers can run, inspect, and contribute to.

Whether you’re looking for a powerful tool to run behind your firewall or want to shape the future of developer tooling, Semaphore Community Edition is here.

Let’s build it together.

semaphore

Opening up Semaphore

Building the future of CI/CD in the open