

Kafka 컨슈머의 **pause**와 **resume** 기능을 활용한 블루-그린 배포는 특히 국내 기술 블로그와 오픈소스 커뮤니티에서 실무적인 해결책으로 상세히 다뤄지고 있습니다. 구체적인 구현 사례와 포스트는 다음과 같습니다.

1. 우아한형제들(배달의민족) 기술 블로그 사례

가장 대표적인 사례로, 우아한형제들에서는 "카프카 컨슈머의 **pause**와 **resume**을 활용한 무중단 배포" 전략을 상세히 공유하고 있습니다 [1].

- 핵심 로직: 컨슈머를 완전히 종료하지 않고 `MessageListenerContainer`의 `pause()` 메서드를 호출하여 메시지 폴링만 일시 정지합니다. 이를 통해 컨슈머 그룹의 멤버 자격을 유지하면서 리밸런싱을 방지합니다 [1].
- 구현 특징: 스프링 프레임워크 환경에서 컨슈머와 컨슈머 그룹, 파티션 구조에 대한 이해를 바탕으로 작성되었으며, 전체 소스코드는 [GitHub](#) 저장소를 통해 공개되어 있습니다 [1].
- 이점: 외부 시스템(DB 등)의 부하를 고려하여 컨슈머 수를 동적으로 조절하거나, 배포 시점에 파티션 소유권을 유지한 채로 안전하게 전환할 수 있습니다 [1].

2. Spring Boot Actuator 기반 자동화 사례

많은 엔지니어들이 **Spring Boot Actuator**의 `/actuator/bindings` 엔드포인트를 외부 스위치로 사용하는 방식을 포스팅하고 있습니다 ``.

- 패턴: `spring-boot-starter-actuator`를 의존성에 추가하고 설정을 통해 `bindings` 엔드포인트를 노출합니다 ``.
- 작동 방식: 외부 배포 도구(예: `Flagger`, `Argo Rollouts`)가 블루 환경의 파드에 `curl -X POST.../pause` 요청을 보내 소비를 멈추고, 그린 환경에 `resume` 신호를 보내는 방식입니다 ``.
- 포스트 키워드: "Pause/Resume Kafka Consumer using Actuator in Spring Cloud Stream"과 같은 제목의 포스트들에서 해당 `YAML` 설정과 호출 시나리오를 확인할 수 있습니다 ``.

3. Redis Pub/Sub을 이용한 다수 파드 동기화 사례

Kubernetes의 여러 컨슈머 파드에 동시에 신호를 전달하기 위해 **Redis**를 메시지 브로커로 활용하는 패턴입니다 [2].

- 구현: **Redis**의 `Pub/Sub` 기능을 사용하여 모든 활성 컨슈머 노드에 `pause/resume` 요청을 전파합니다. 이는 단일 `REST` 호출이 특정 파드 하나만 제어하는 한계를 극복하기 위해 사용됩니다 ``.
- 관련 도구: `Backbeat`와 같은 데이터 복제 엔진의 디자인에서 이러한 **Redis** 기반의 컨슈머 제어 방식이 구현 사례로 언급됩니다 [2].

4. Flagger 및 Argo Rollouts 연동 사례

배포 도구의 웹훅(**Webhook**) 기능을 이용해 위에서 언급한 `pause/resume API`를 호출하는 파이프라인 구성 사례입니다.

- **Flagger**: `pre-rollout` 웹훅에서 그린 환경을 검증하고, 성공 시 블루 환경의 컨슈머를

정지시키는 스크립트를 실행하는 구조입니다 ``.

- **Argo Rollouts:** AnalysisTemplate을 통해 Consumer Lag 지표를 Prometheus로 관찰하며, 배포 단계 사이사이에 AnalysisRun이 웹후크를 호출하여 컨슈머 상태를 제어합니다 ``.

이러한 포스트들은 주로 "**Kafka Zero-downtime Deployment**", "**Spring Kafka Pause Resume Actuator**" 등의 키워드로 검색 시 기술 블로그(Tistory, Velog, Medium 등)에서 상세한 설정값과 함께 찾아볼 수 있습니다 ``.