Spring Cloud Stream을 활용한
실시간 콘텐츠 배포 시스템 구축기

이경일 / Kakao / 쇼핑플랫폼개발셀

발표자

- 이경일 / leekyoungil@gmail.com
- Kakao 쇼핑플랫폼개발셀
- 커머스플랫폼 상품 도메인 개발담당

발표자

- 이경일 / leekyoungil@gmail.com
- Kakao 쇼핑플랫폼개발셀
- 커머스플랫폼 상품 도메인 개발담당

kakao farmer Jeju

KAKAO 메이커스 위드 카카오

발표자

- 이경일 / leekyoungil@gmail.com
- Kakao 쇼핑플랫폼개발셀
- 커머스플랫폼 상품 도메인 개발담당
- 잘하는 것은 없지만 열심히 한다고 생각하는 노가다 전문 개발자

발표에 앞서

- 대상(초급) : 강연 주제에 대해 아무것도 모르는 개발자

발표에 앞서

- 대상(초급) : 강연 주제에 대해 아무것도 모르는 개발자
- Spring Cloud에 관해 들어보지 못했거나 아직 사용은 해보지 못한 개발자

콘텐츠란?

콘텐츠란?

콘텐츠란?

- 미리 가공해둔 정적인 데이터 파일을 의미함

콘텐츠 배포 시스템이 왜 필요한가요?

콘텐츠 배포 시스템이 왜 필요한가요?

- PV가 높고 콘텐츠 변화가 잦은 서비스

콘텐츠 배포 시스템이 왜 필요한가요?

- PV가 높고 콘텐츠 변화가 잦은 서비스
- Cache Layer가 필요함

콘텐츠 배포 시스템이 왜 필요한가요?

- PV가 높고 콘텐츠 변화가 잦은 서비스
- Cache Layer가 필요함
- PV증가에 따른 Cache Server의 증가

콘텐츠 배포 시스템이 왜 필요한가요?

- PV가 높고 콘텐츠 변화가 잦은 서비스
- Cache Layer가 필요함
- PV증가에 따른 Cache Server의 증가
- 관리 포인트가 늘어남

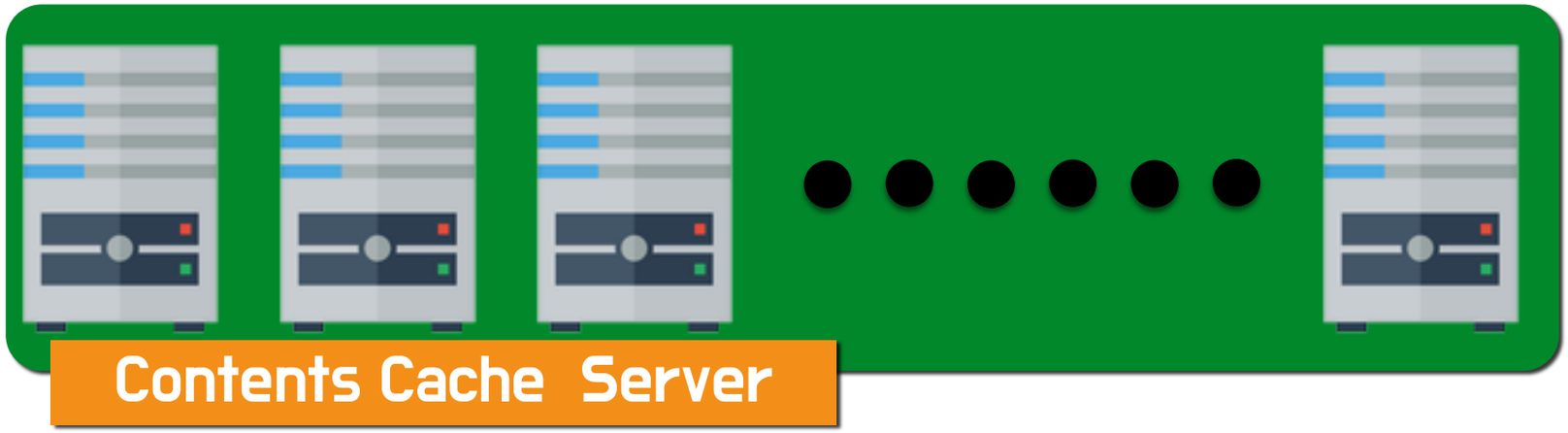
콘텐츠 배포 시스템이 왜 필요한가요?

- PV가 높고 콘텐츠 변화가 잦은 서비스
- Cache Layer가 필요함
- PV증가에 따른 Cache Server의 증가
- 관리 포인트가 늘어남

- Cache is Cash

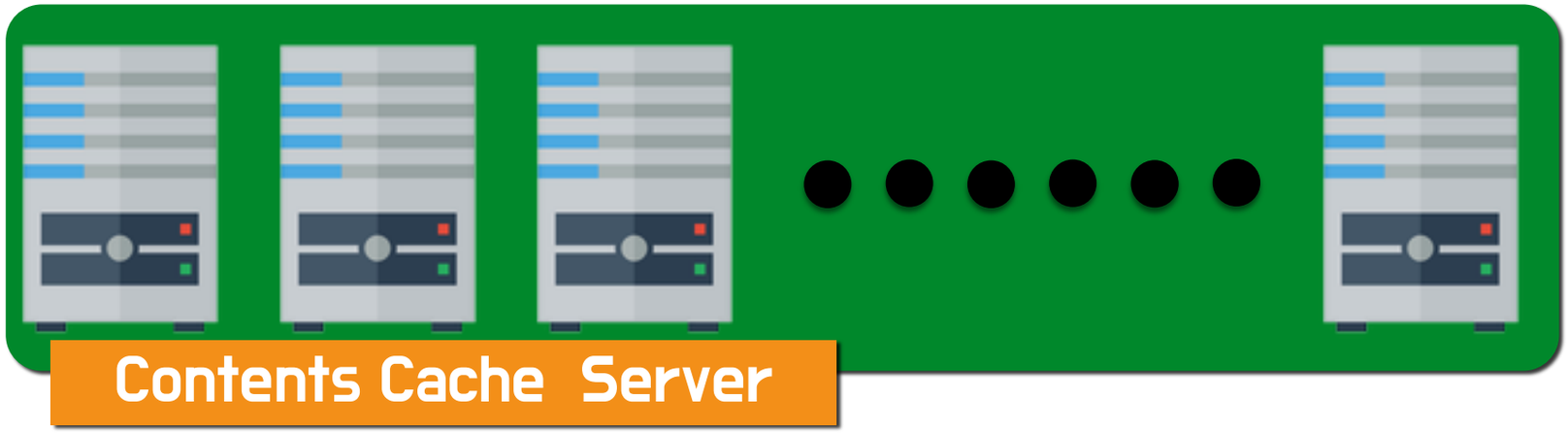
문제점의 발견

- 대용량 트래픽 처리를 위한 Cache Server



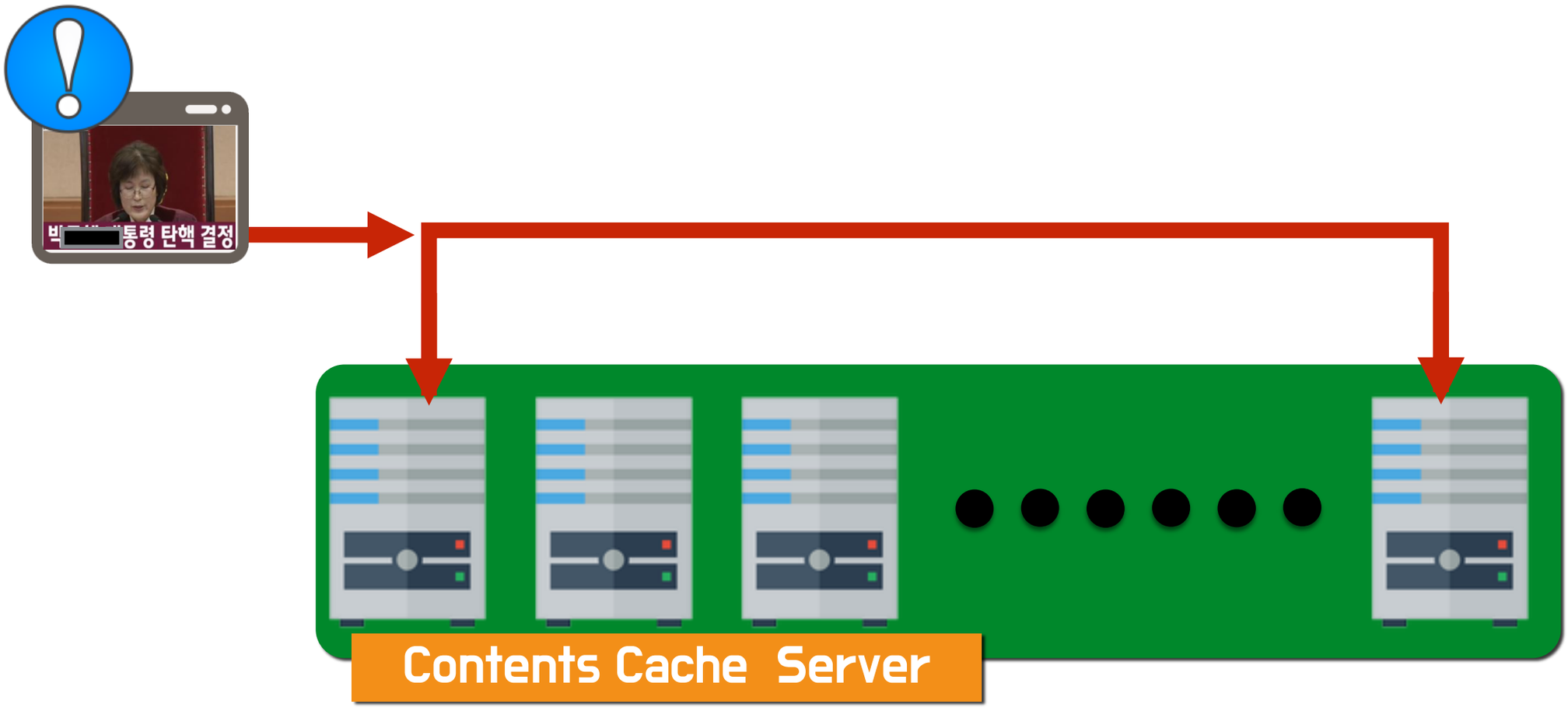
문제점의 발견

- 신규 콘텐츠 발행 시



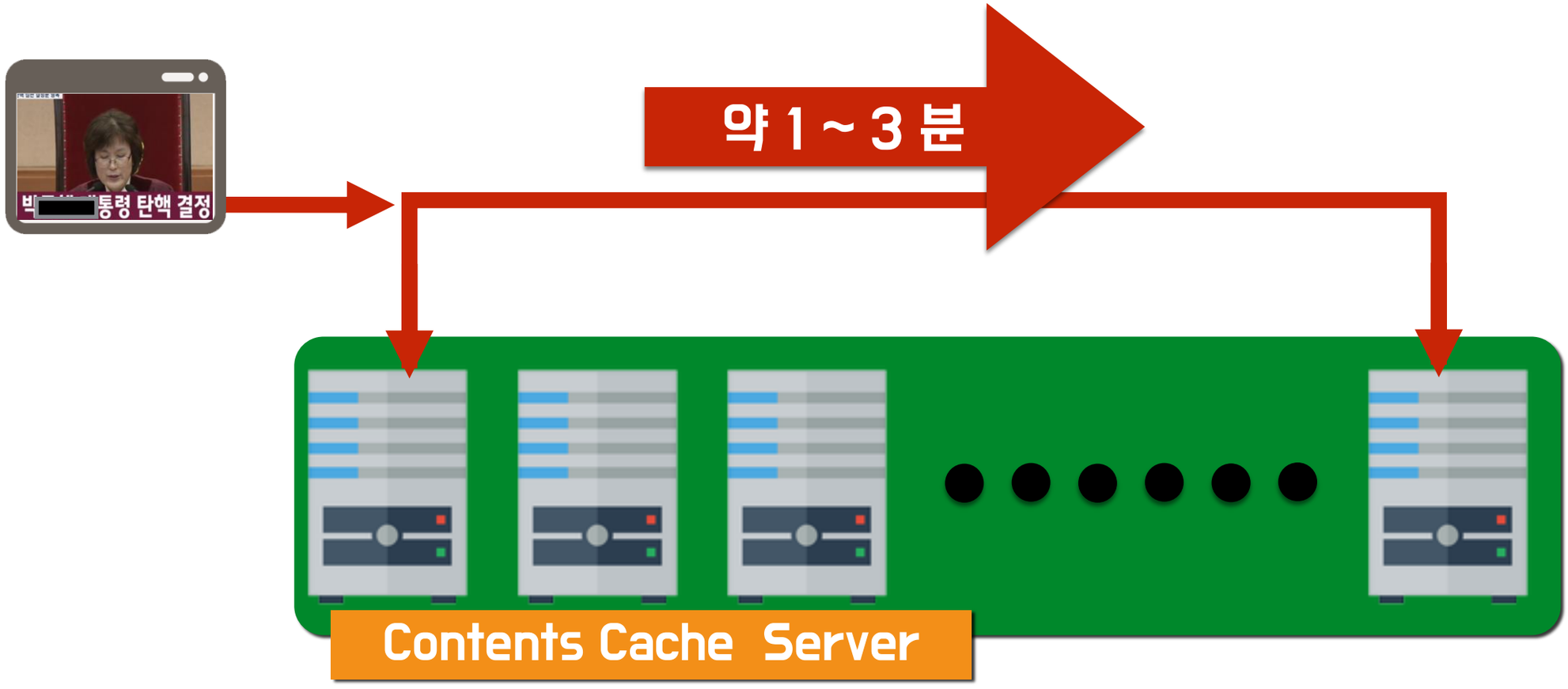
문제점의 발견

- 콘텐츠 Cache Server에 배포됨



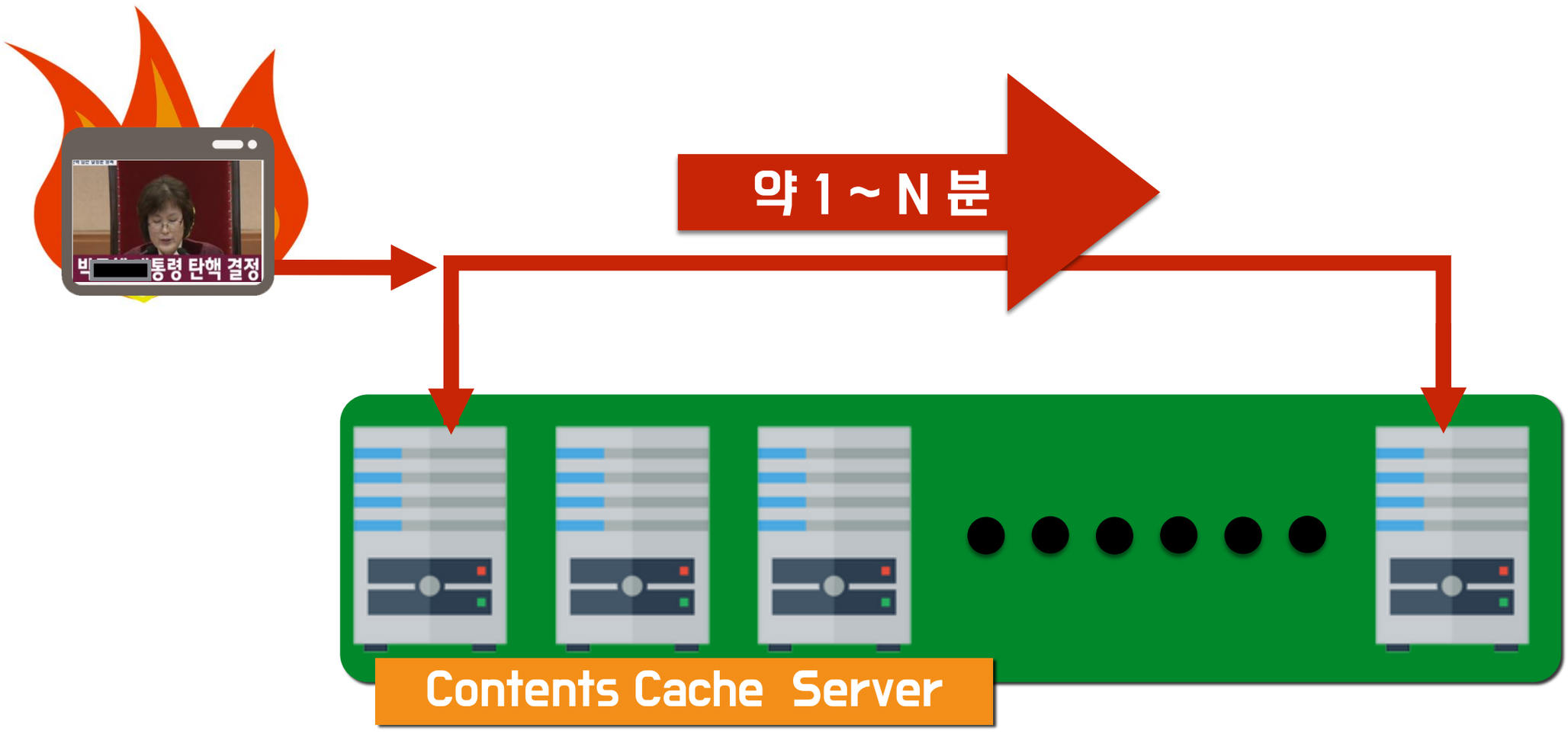
문제점의 발견

- 콘텐츠 Cache Server에 배포됨



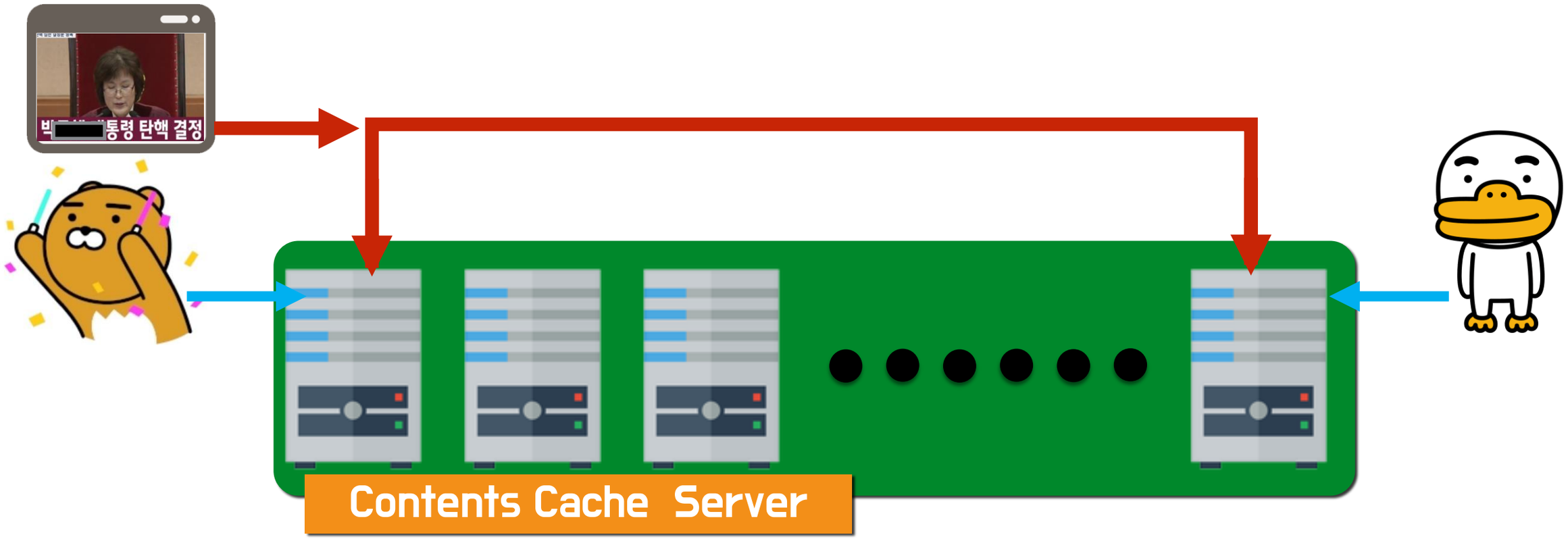
문제점의 발견

- Cache Server가 늘어날수록 시간이 오래걸림



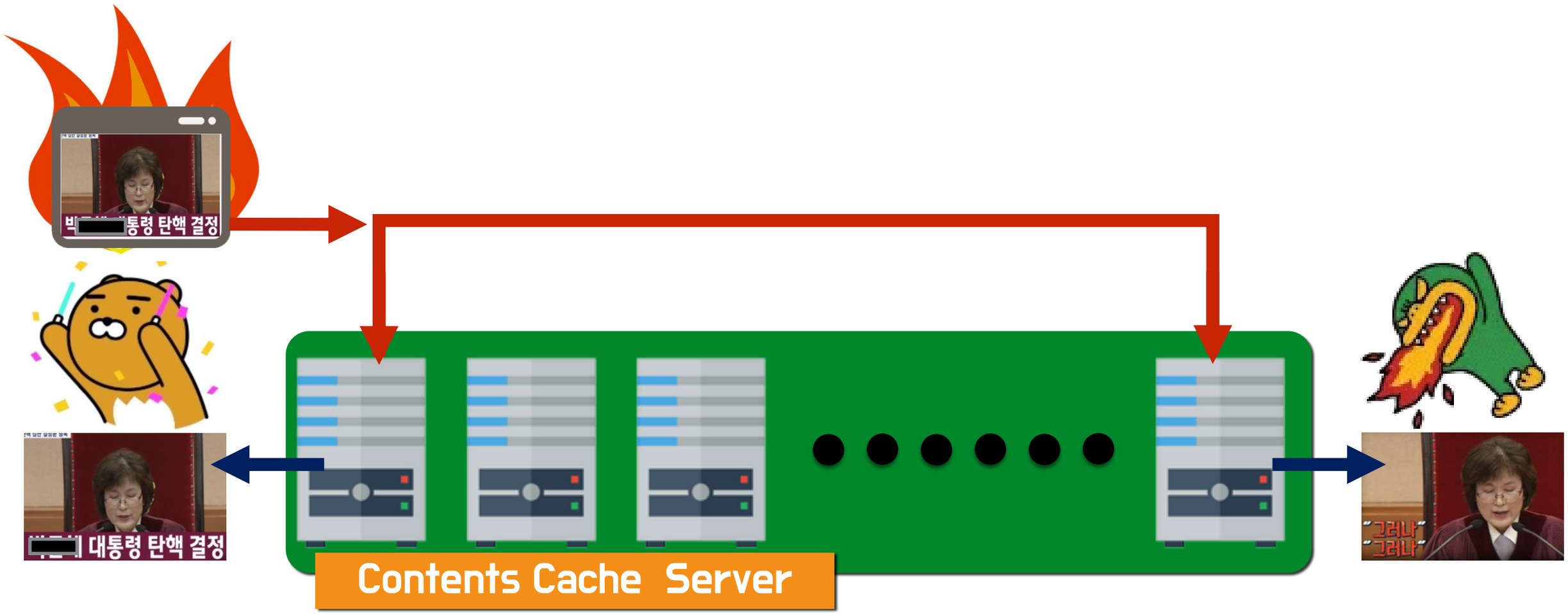
문제점의 발견

- 또다른 문제...



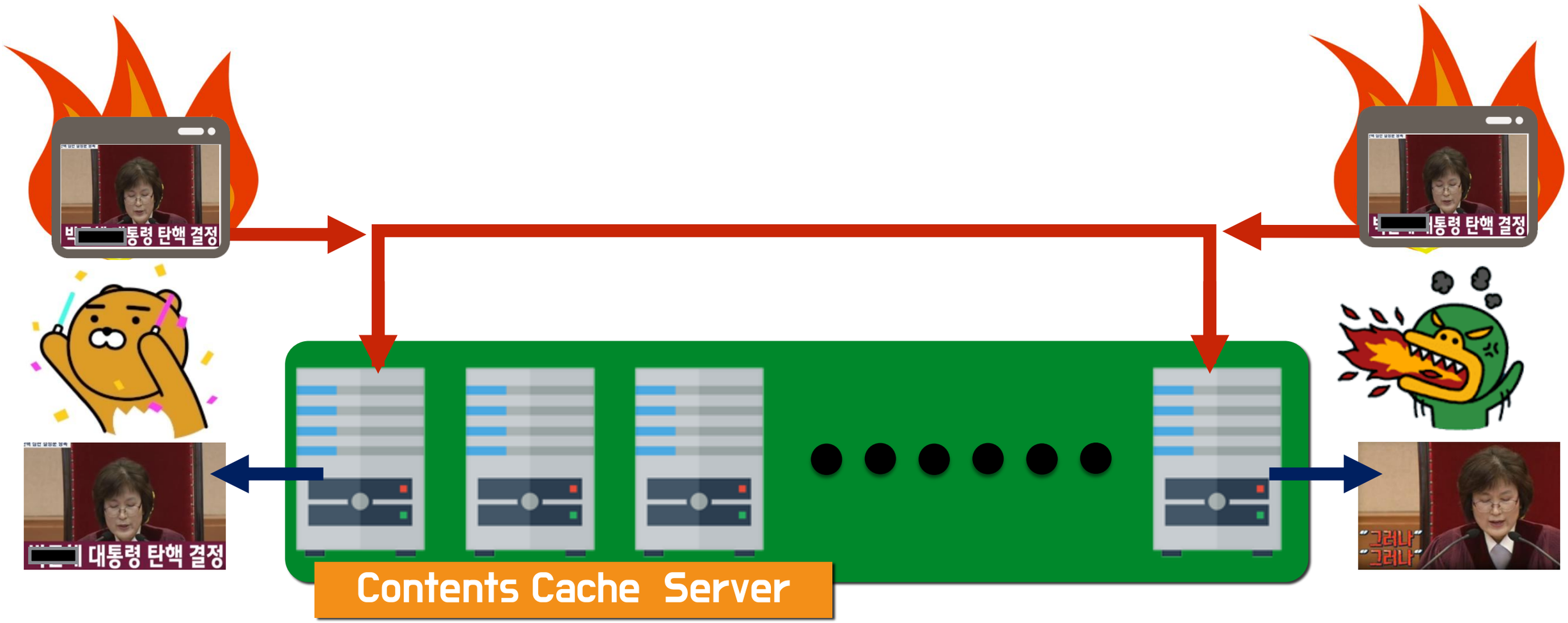
문제점의 발견

- Cache Server간 콘텐츠의 동기화 문제



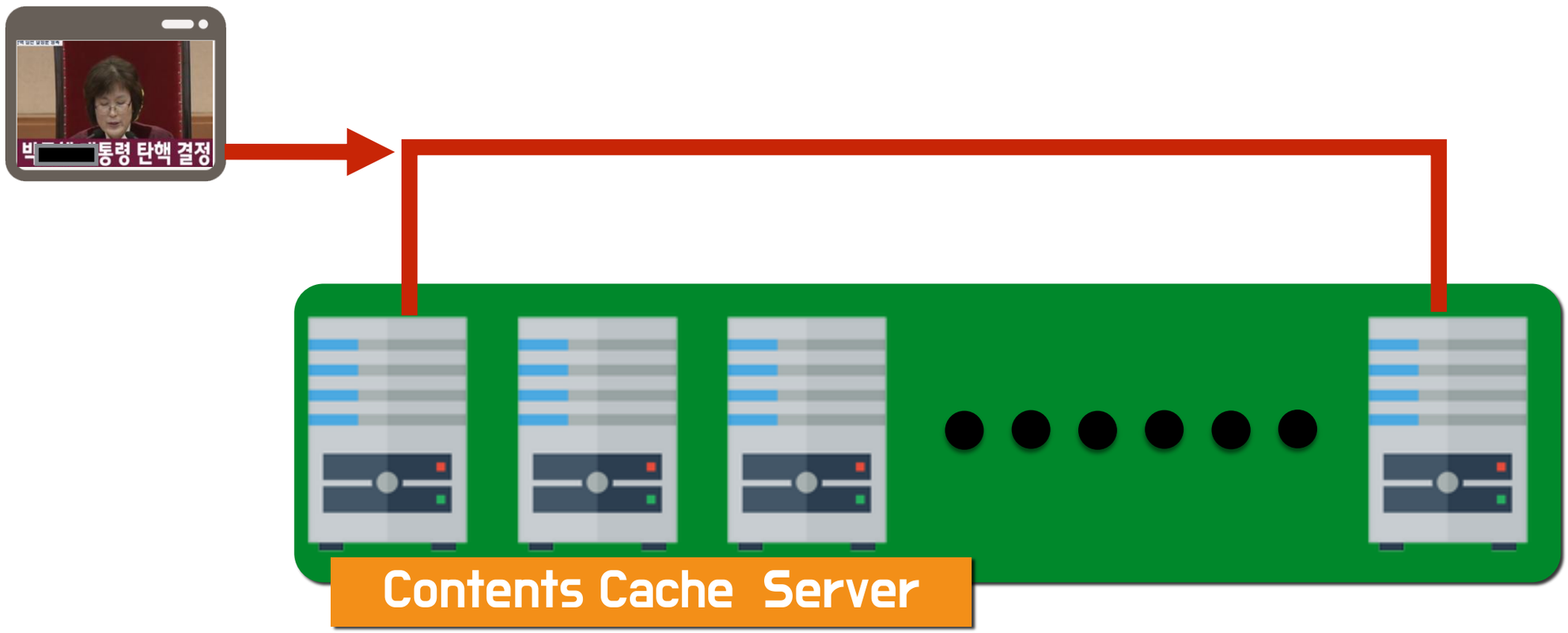
어떻게 해결할 것인가?

- 콘텐츠 배포서버의 증설? (X)



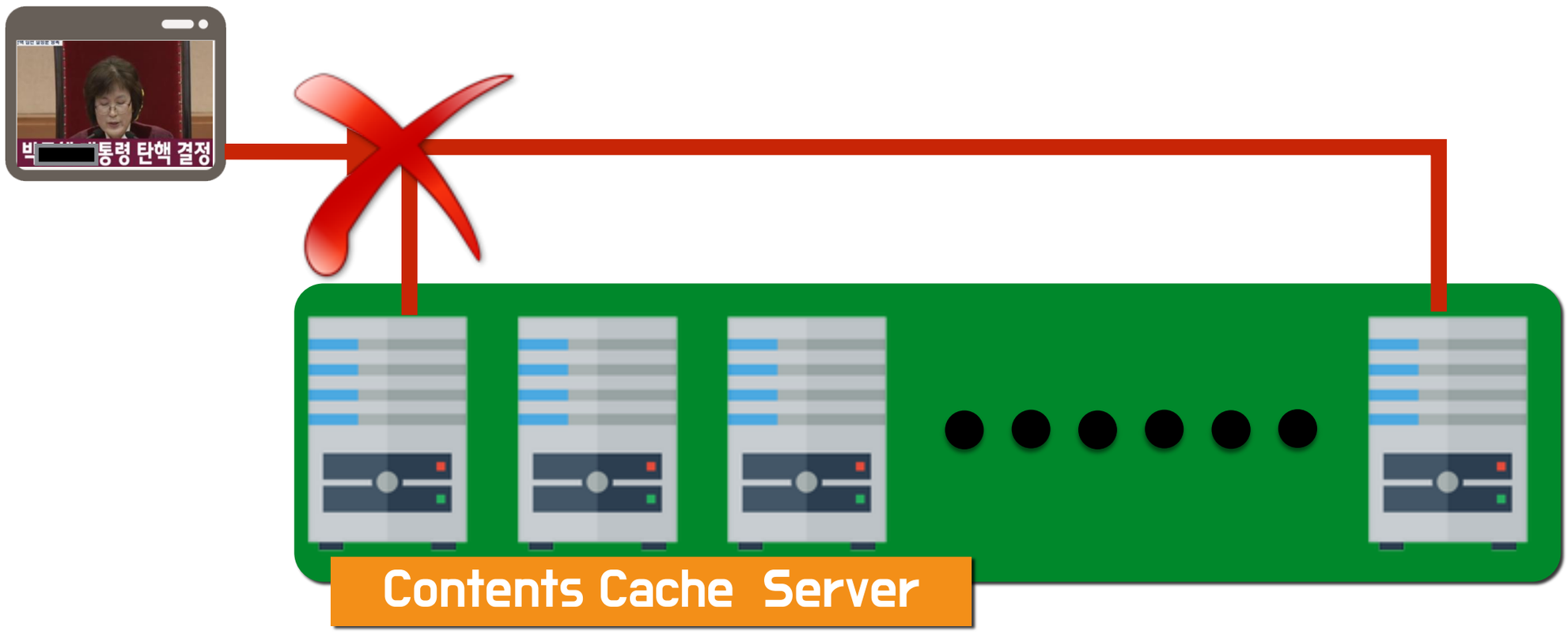
어떻게 해결할 것인가?

- 콘텐츠 배포의 패러다임을 바꿈



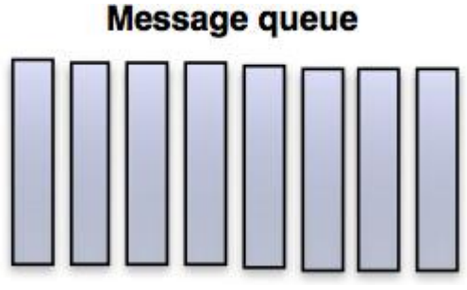
어떻게 해결할 것인가?

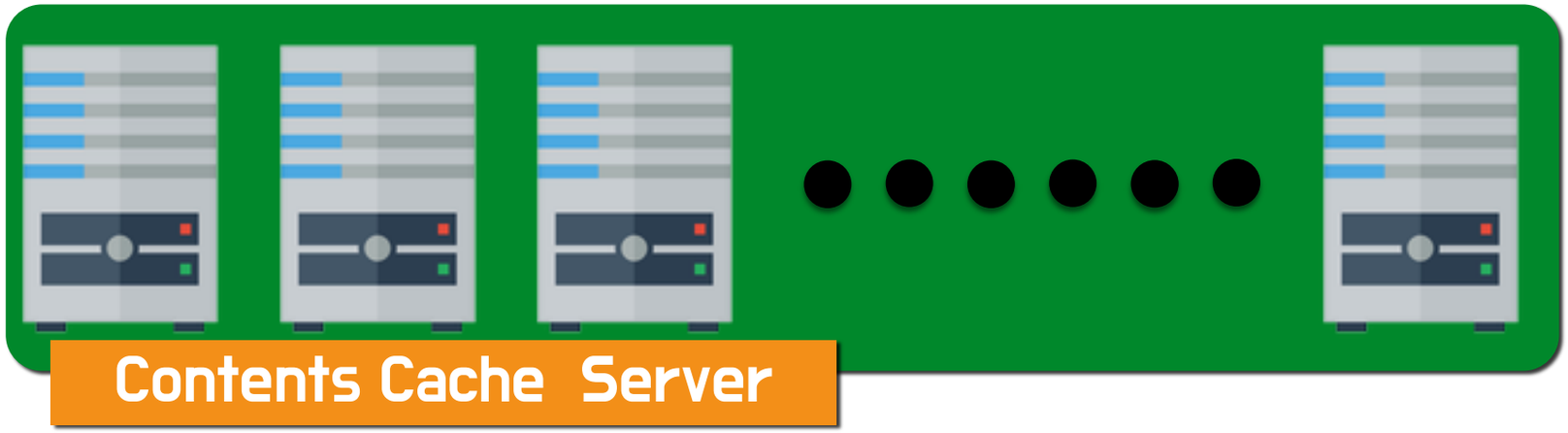
- 콘텐츠 배포의 패러다임을 바꿈



어떻게 해결할 것인가?

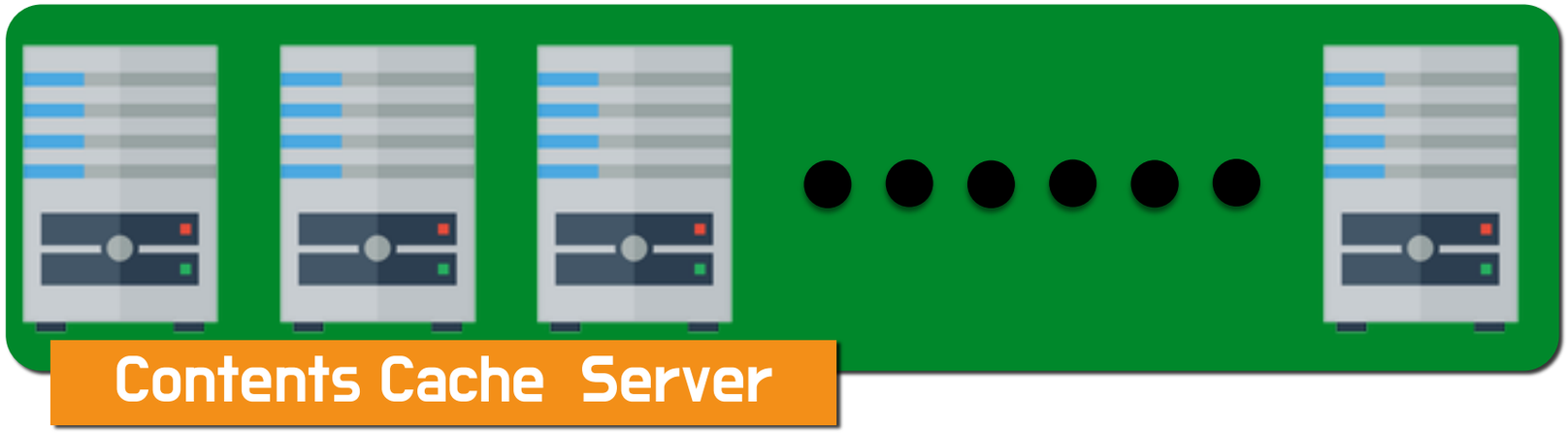
- 콘텐츠 배포의 패러다임을 바꿈





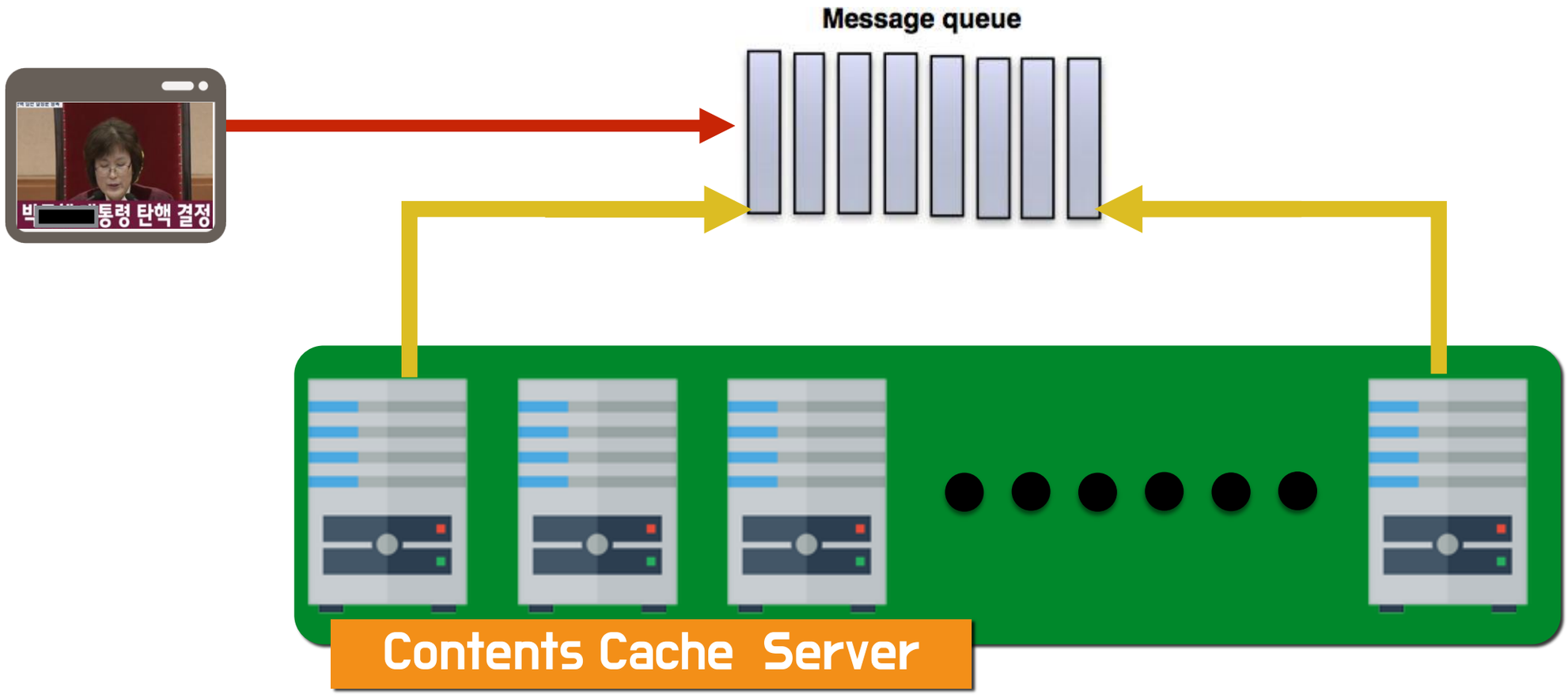
어떻게 해결할 것인가?

- 콘텐츠 배포의 패러다임을 바꿈



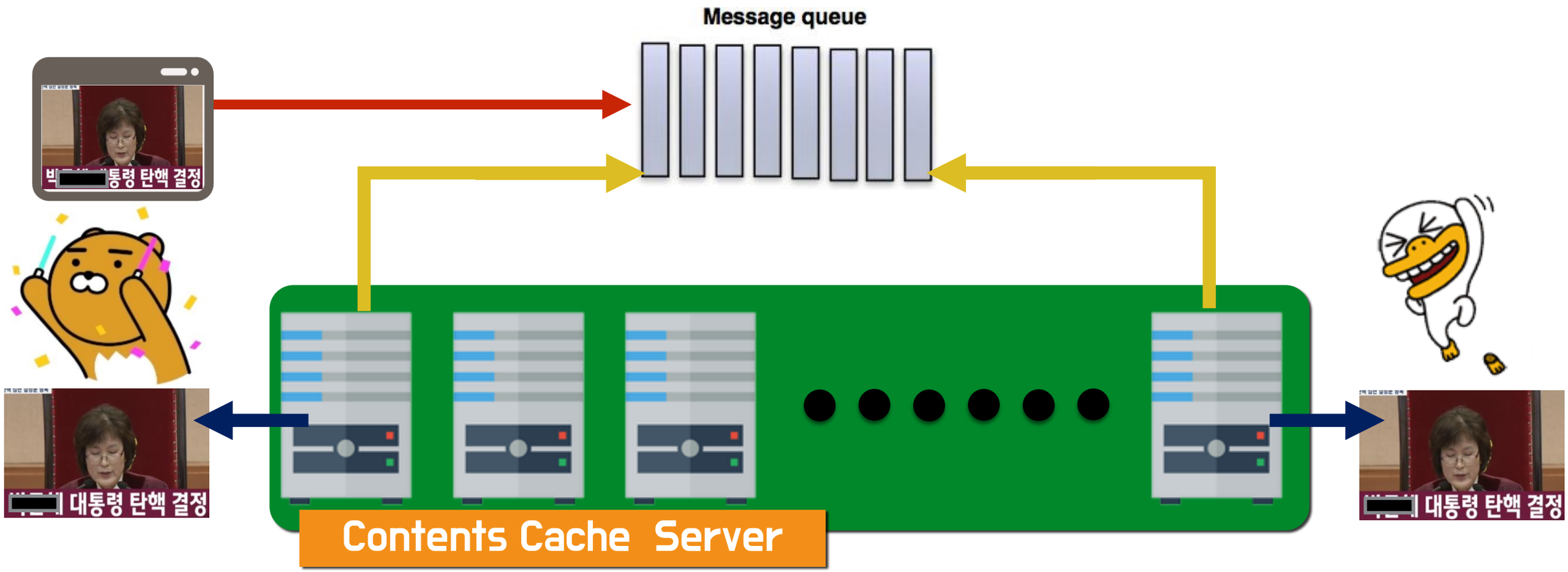
어떻게 해결할 것인가?

- 콘텐츠 배포의 패러다임을 바꿈



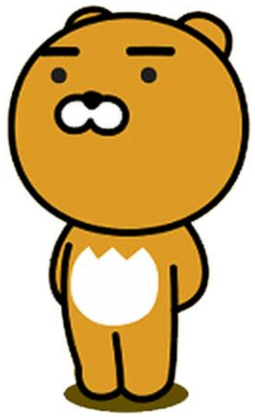
어떻게 해결할 것인가?

- 콘텐츠 배포의 패러다임을 바꿈 (O)



정리를 하자면 - 기존에는

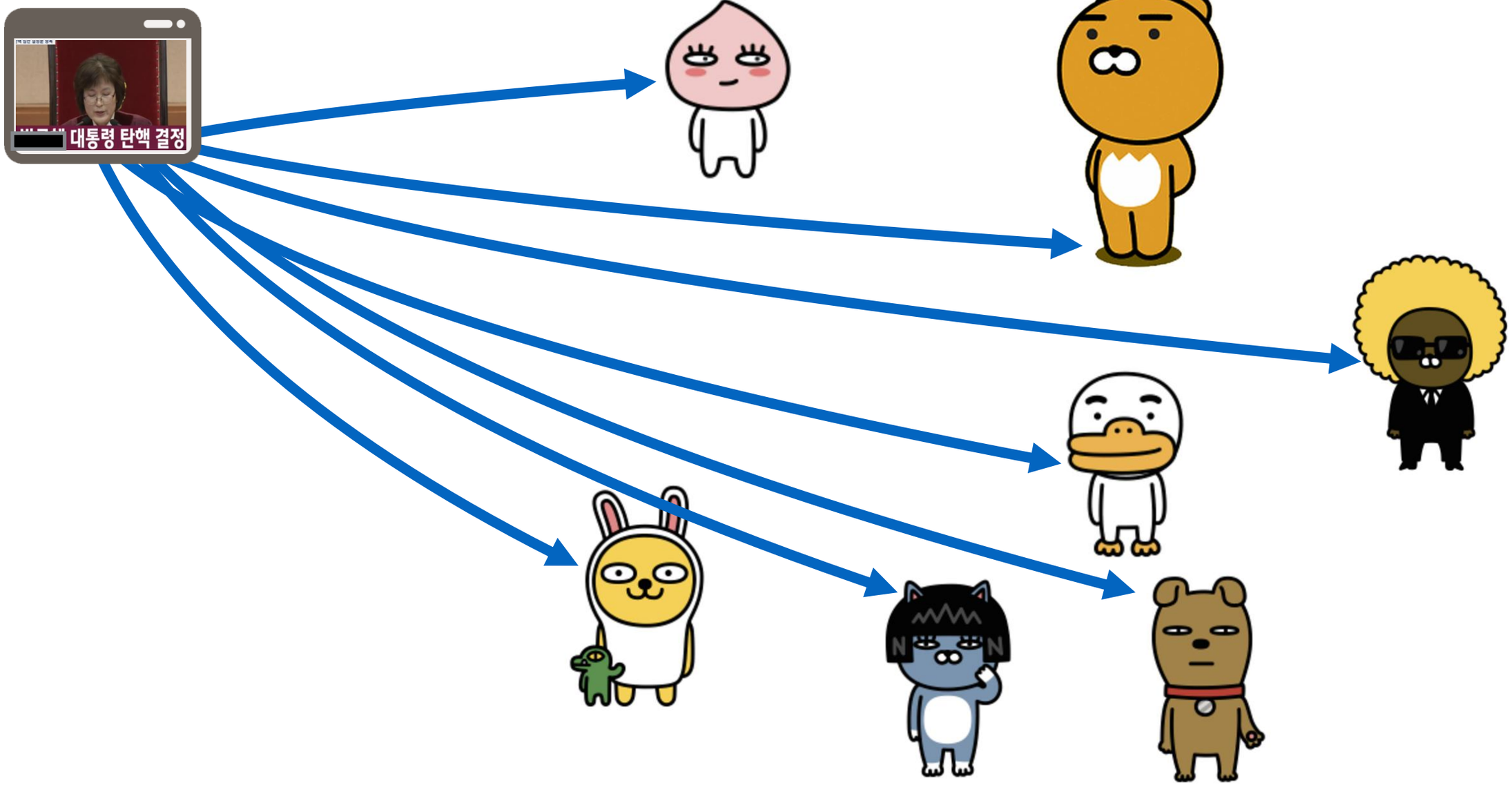
- 신규 콘텐츠 발행



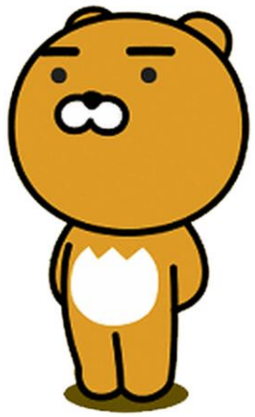
kakao

정리를 하자면 - 기존에는

- 신규 콘텐츠 발행 -> Push



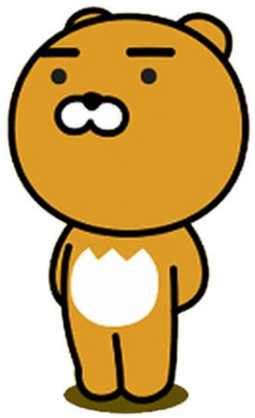
정리를 하자면 - 새로운 방식은



kakao

정리를 하자면 - 새로운 방식은

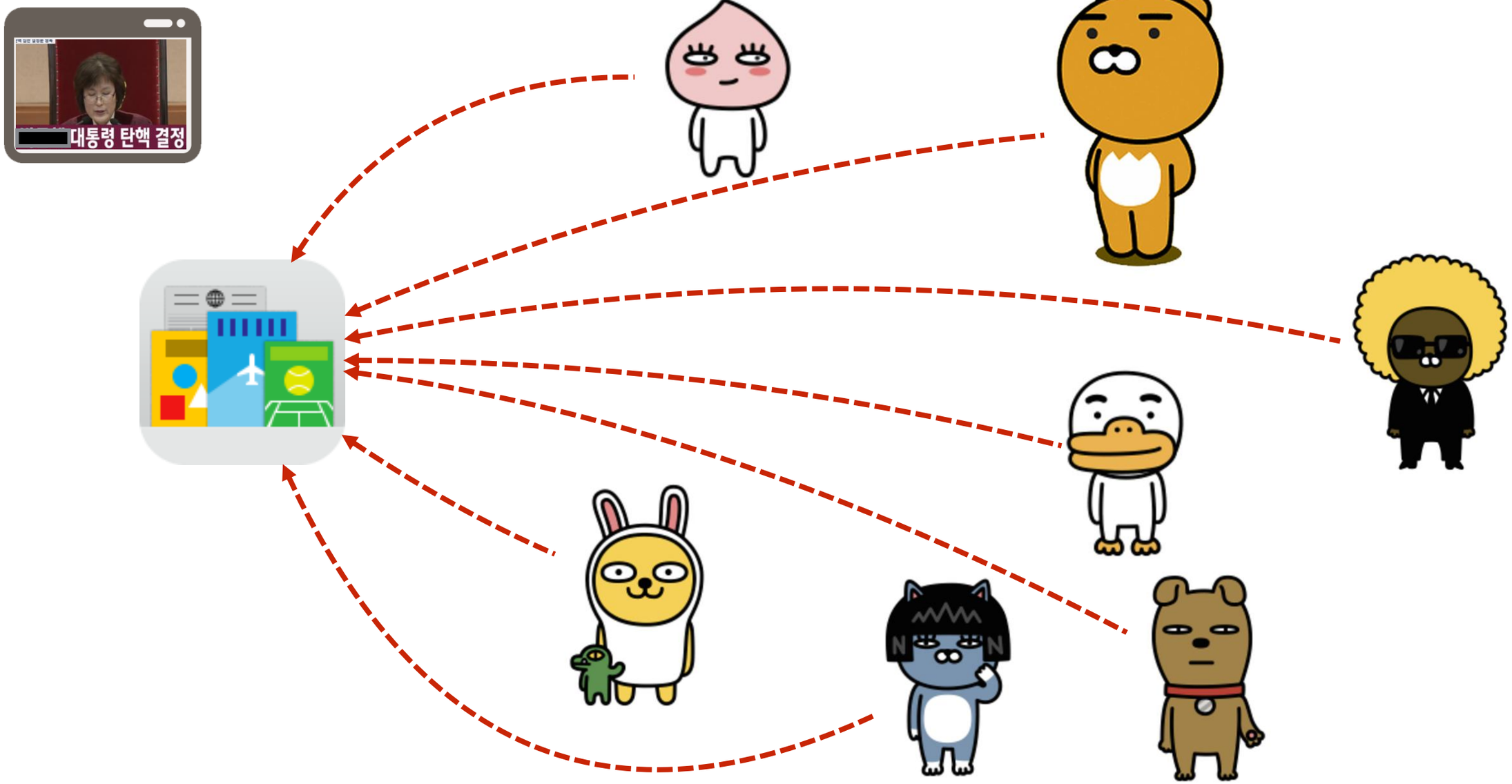
- 중간에 Event Store생성



kakao

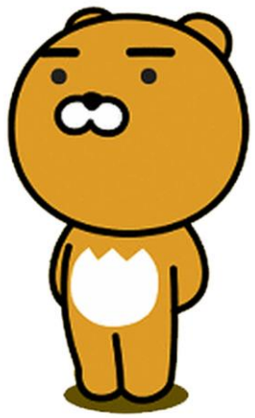
정리를 하자면 - 새로운 방식은

- 구독자는 이 Event Store를 보고 있음



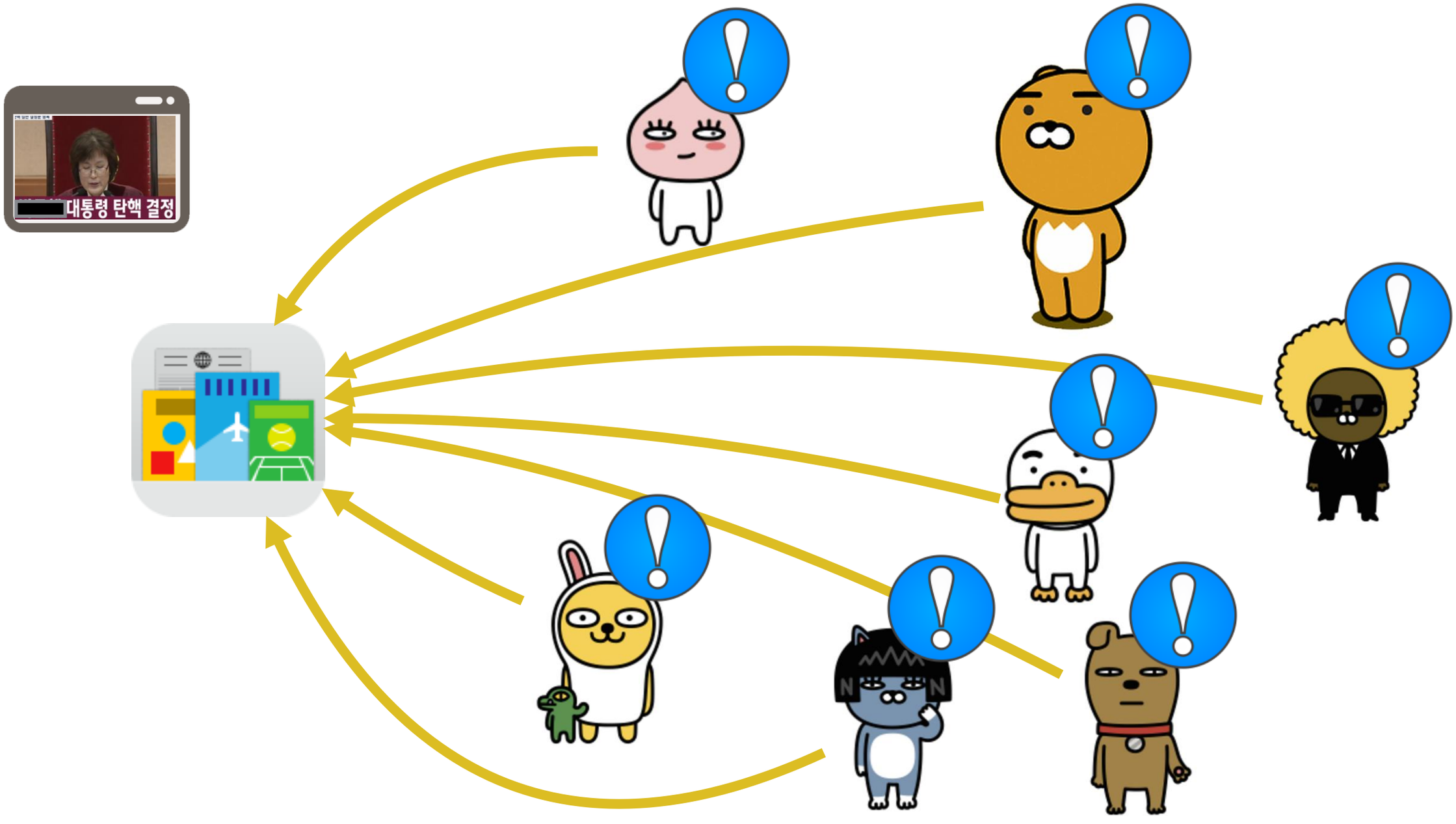
정리를 하자면 - 새로운 방식은

- 신규 콘텐츠 발행 -> Event Store에 저장



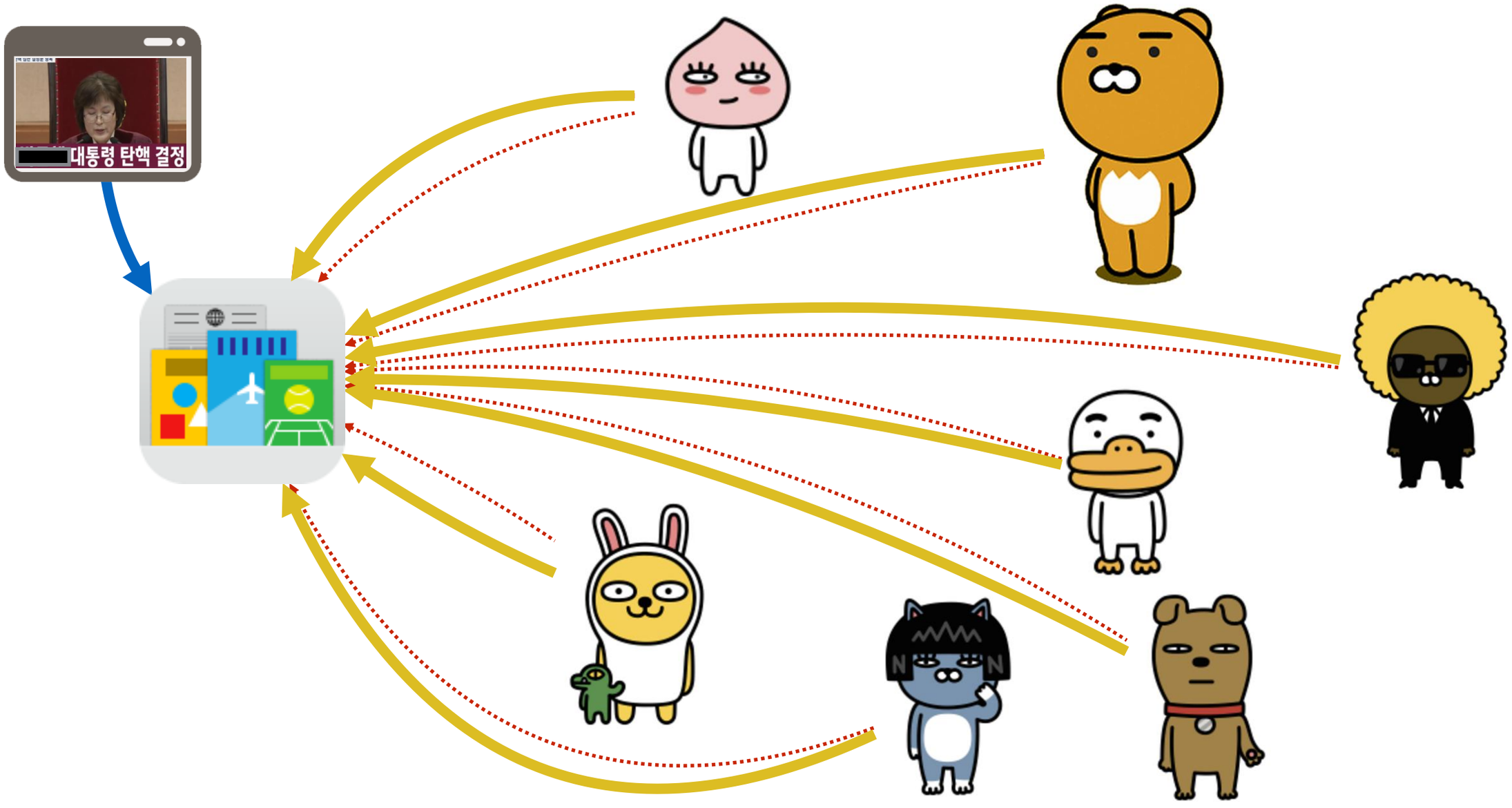
정리를 하자면 - 새로운 방식은

- 구독자는 콘텐츠를 Event Store에서 구독함

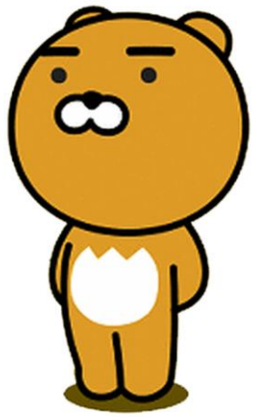


Pub/Sub Pattern

- 발행/구독 - 중간에 Event Store를 둠



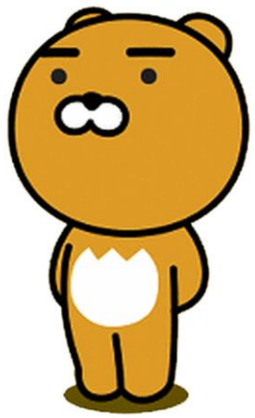
성능적으로 보면 - 기존에는



kakao

성능적으로 보면 - 기존에는

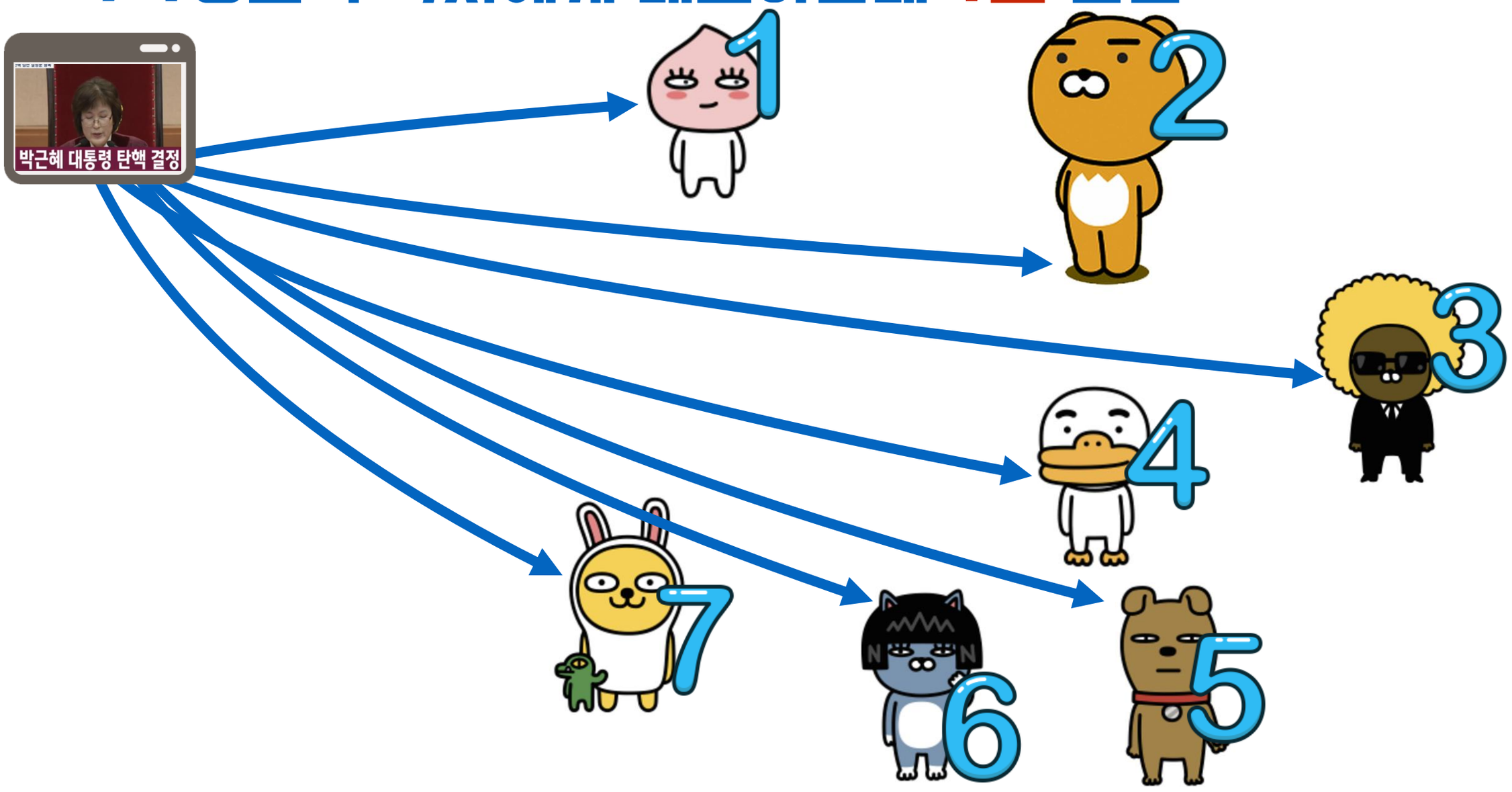
- Push 서버의 성능을 1TPS라 하면



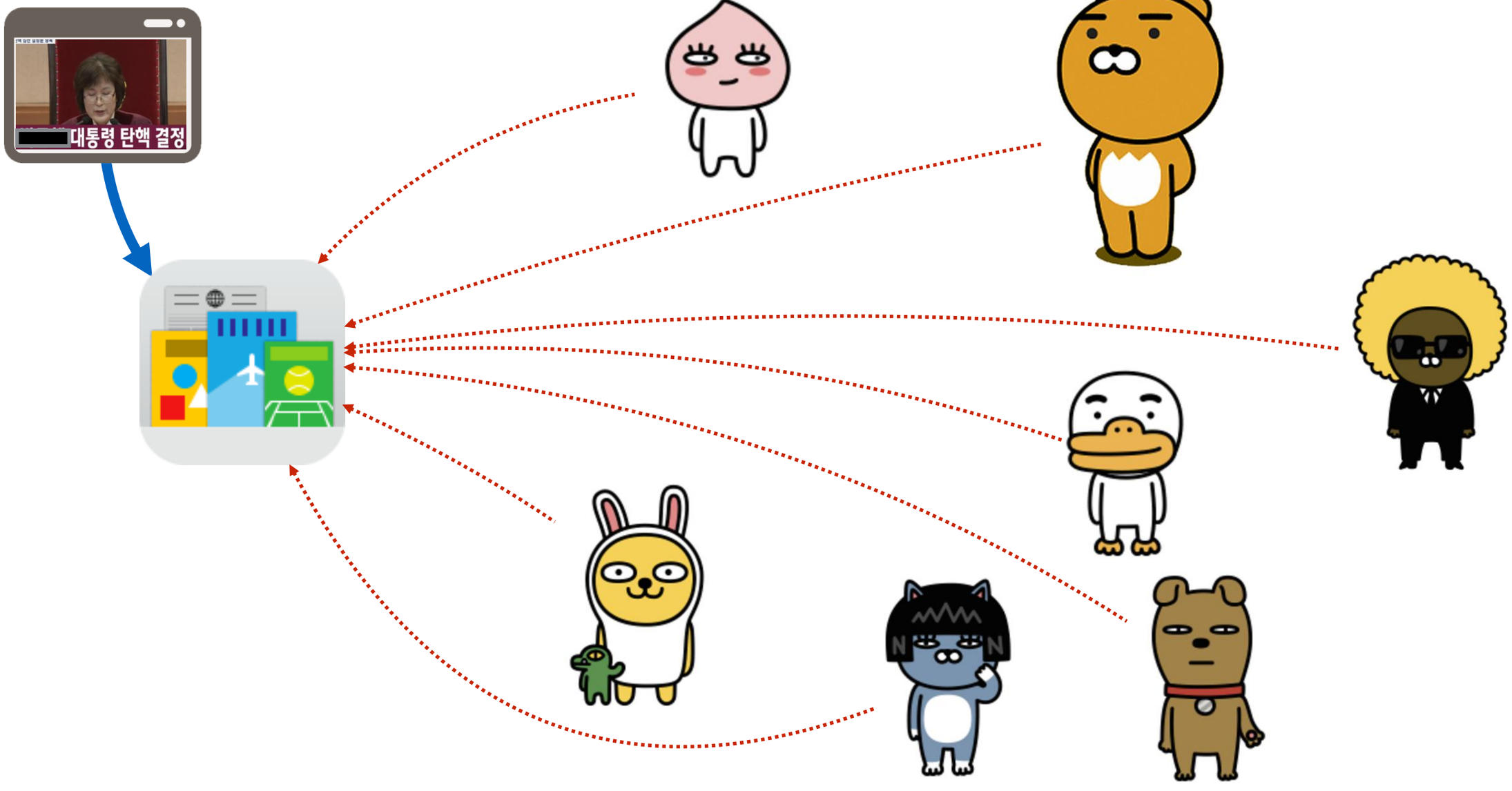
성능적으로 보면 - 기존에는

- Push 서버의 성능을 1TPS라 하면

  -> 7명의 구독자에게 배포하는데 7초 걸림

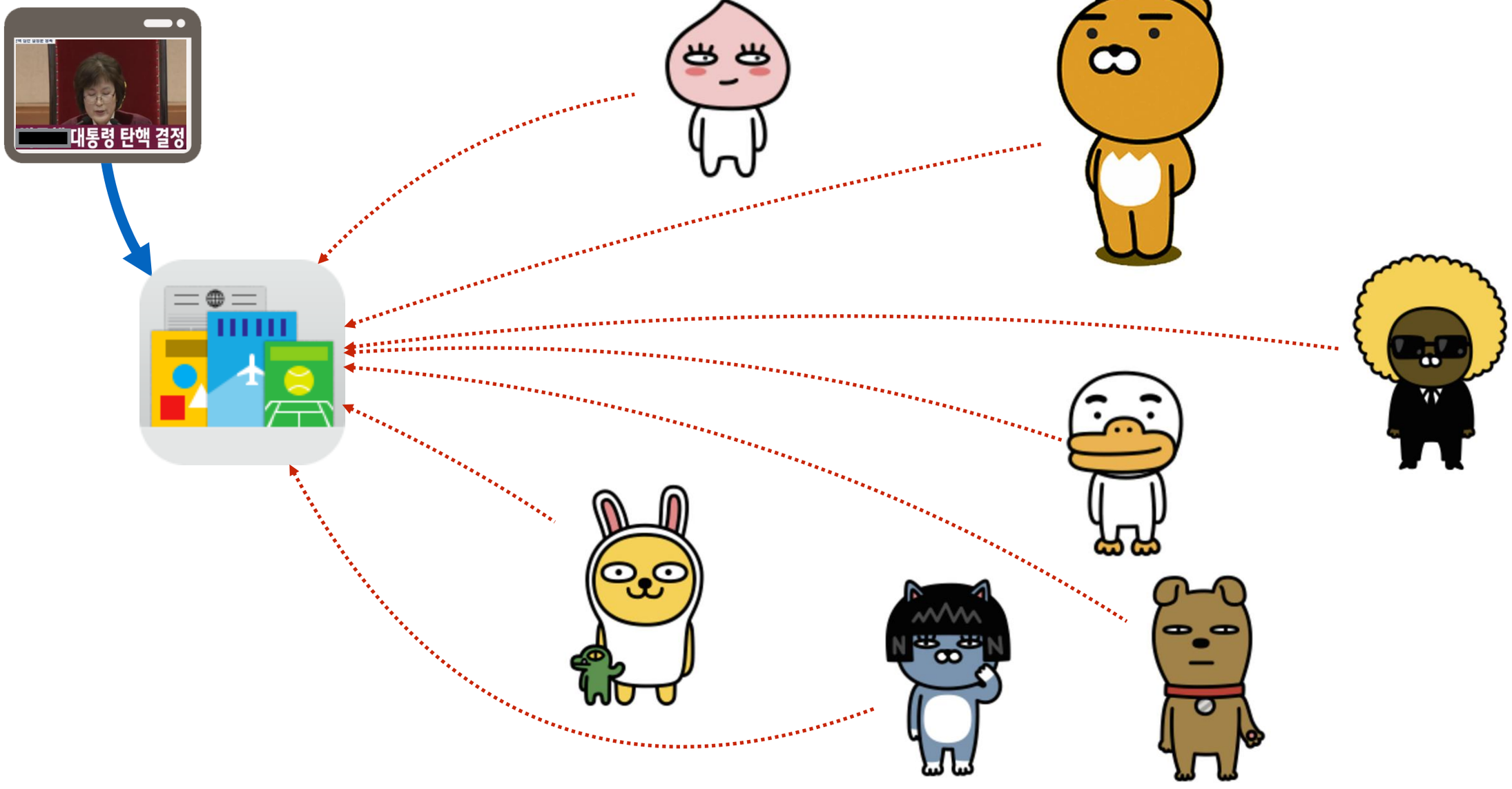


성능적으로 보면 - 새로운 방식은



성능적으로 보면 - 새로운 방식은

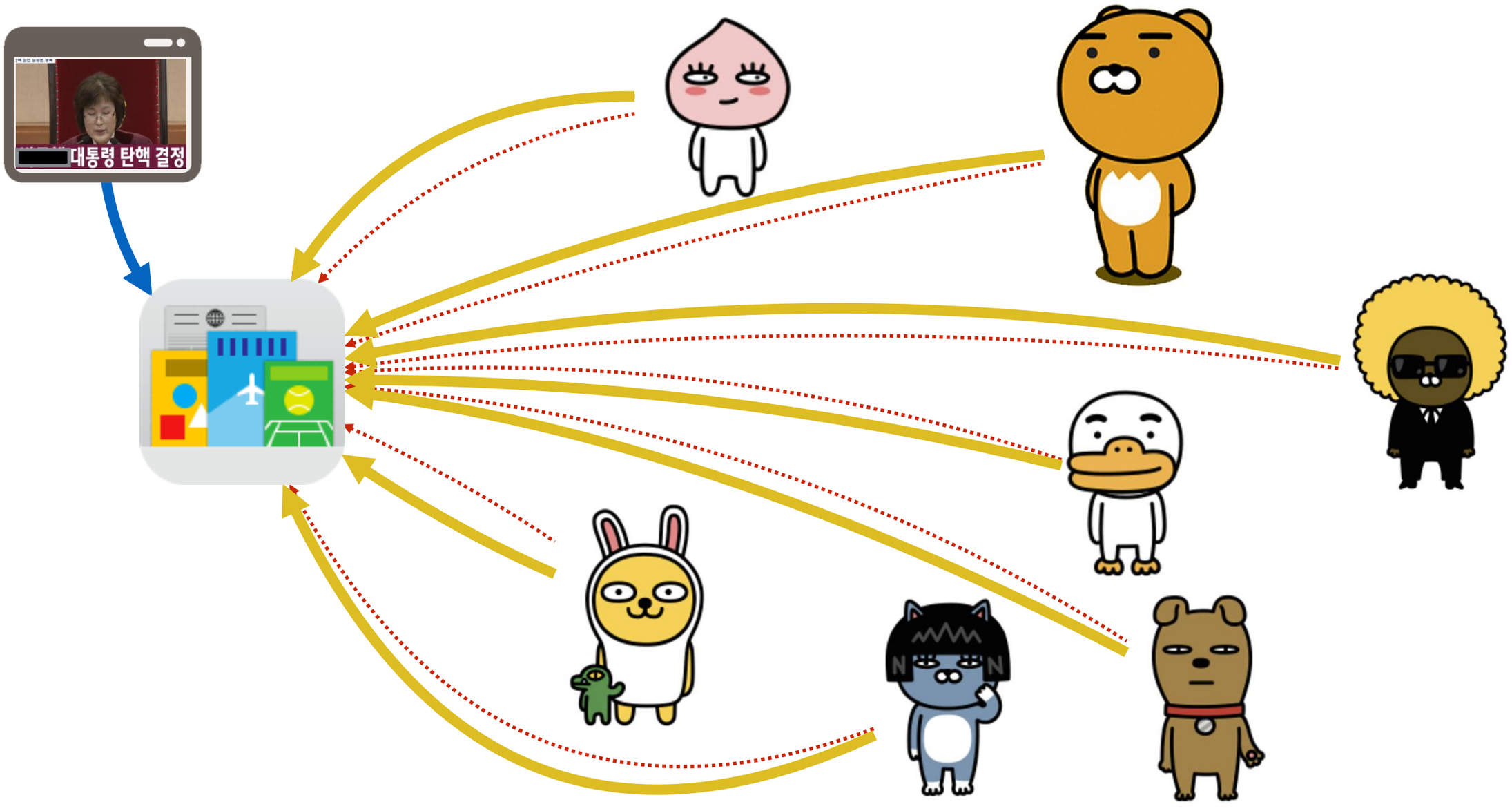
- 각 **구독자**의 성능을 **1TPS**라 하면



성능적으로 보면 - 새로운 방식은

- 각 구독자의 성능을 1TPS라 하면

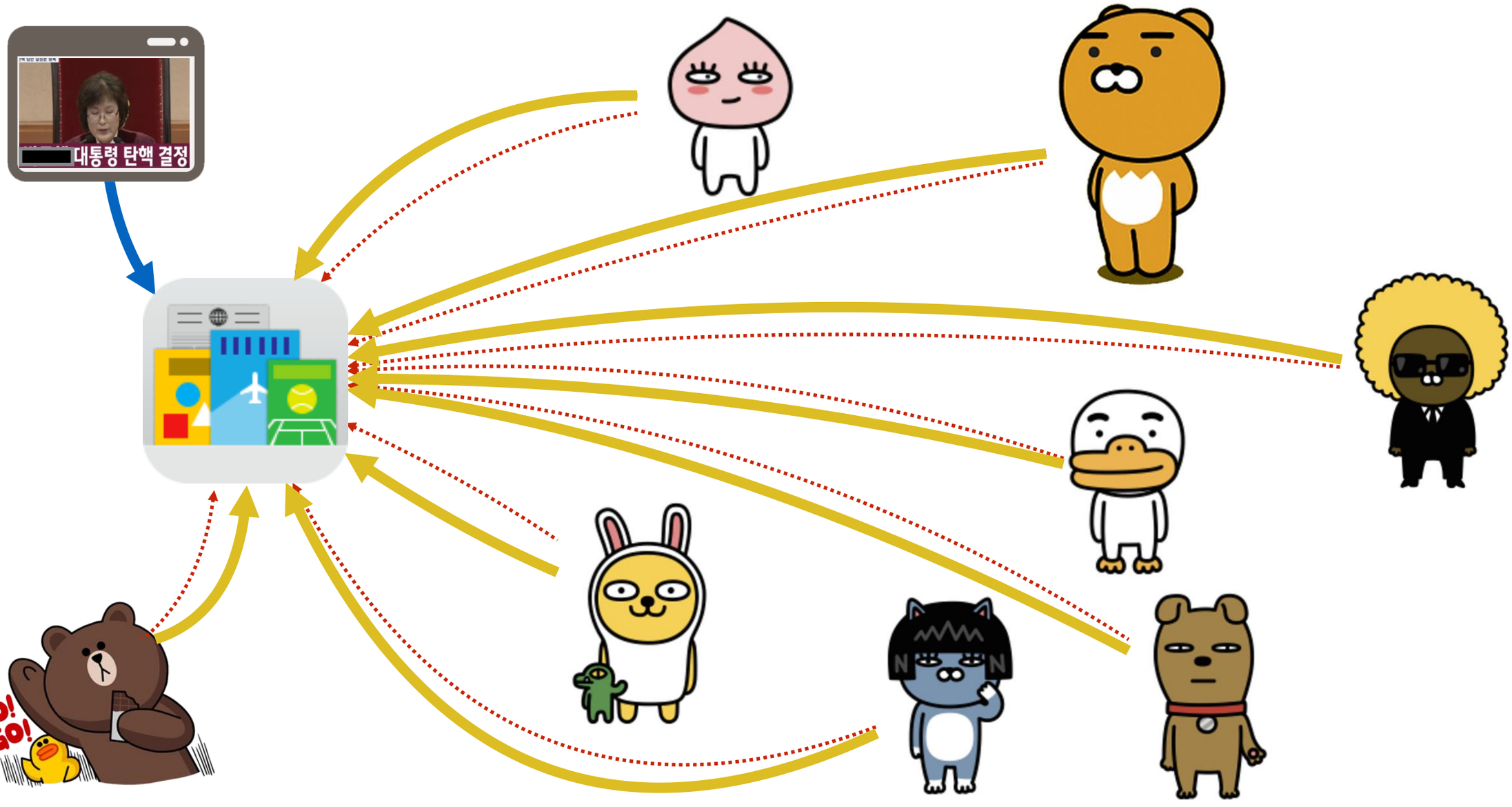
  -> 7명의 구독자가 구독하는데 1초 걸림



성능적으로 보면 - 새로운 방식은

- 각 구독자의 성능을 1TPS라 하면

  -> 8명의 구독자가 구독하는데 1초 걸림



키워드

키워드

- Spring Cloud Stream

키워드

- Spring Cloud Stream
- Spring Cloud & Spring Boot

키워드

- Spring Cloud Stream
- Spring Cloud & Spring Boot
- Stream & Batch

키워드

- Spring Cloud Stream
- Spring Cloud & Spring Boot
- Stream & Batch
- Pub/Sub - Publish & Subscribe

키워드

- Spring Cloud Stream
- Spring Cloud & Spring Boot
- Stream & Batch
- Pub/Sub - Publish & Subscribe
- Message Queue (Kafka, RabbitMQ)

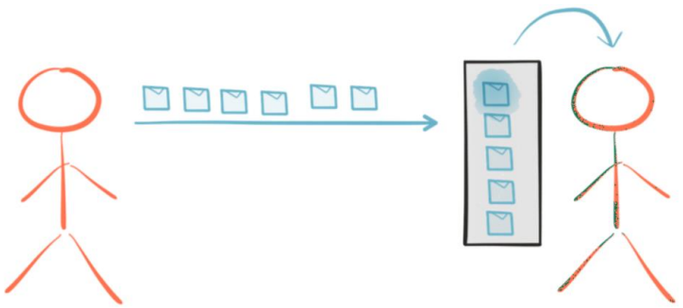
일단 시작 - 삽부터 들고

일단 시작 - 필요한 것은?

- **Message Queue**

일단 시작 - 필요한 것은?

- Message Queue
- 발행-구독



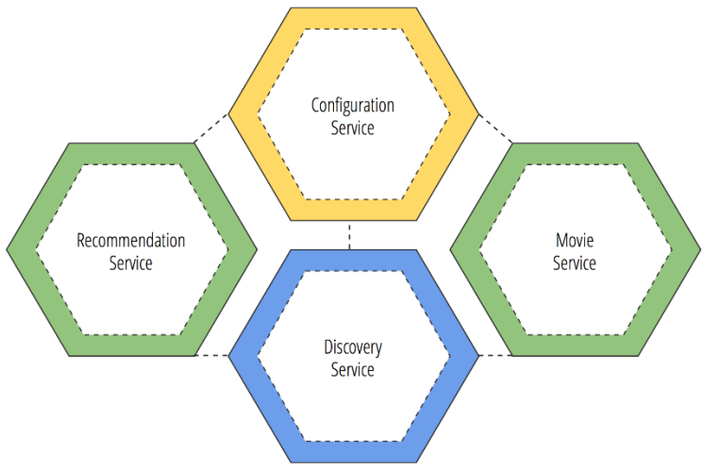
APACHE kafka™
A distributed streaming platform

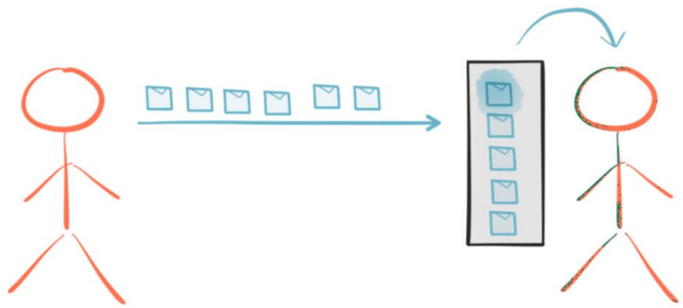
ActiveMQ

RabbitMQ

일단 시작 - 필요한 것은?

- Message Queue
- 발행-구독
- 역할의 분담





APACHE kafka™
A distributed streaming platform

ActiveMQ

RabbitMQ

어떤 것을 쓰는 것이 좋을까?

어떤 것을 쓰는 것이 좋을까?

- JVM - Java

어떤 것을 쓰는 것이 좋을까?

- JVM - Java - Spring

어떤 것을 쓰는 것이 좋을까?

- JVM - Java - Spring Boot

어떤 것을 쓰는 것이 좋을까?

- JVM - Java - Spring Boot
- Message Queue - Kafka (Pub/Sub)

플랫폼의 결정

- JVM - Java - Spring Boot
- Message Queue - Kafka (Pub/Sub)

플랫폼의 결정

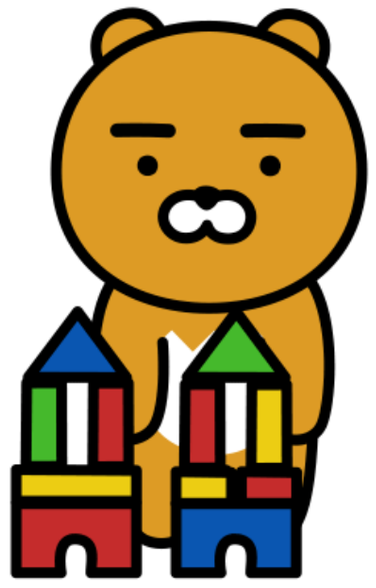
- JVM - Java - Spring Boot
- Message Queue - Kafka (Pub/Sub)
- Spring Cloud Stream

플랫폼의 결정

- JVM - Java - Spring Boot
- Message Queue - Kafka (Pub/Sub)
- Spring Cloud Stream
- 그런데 여기서 잠깐만요.

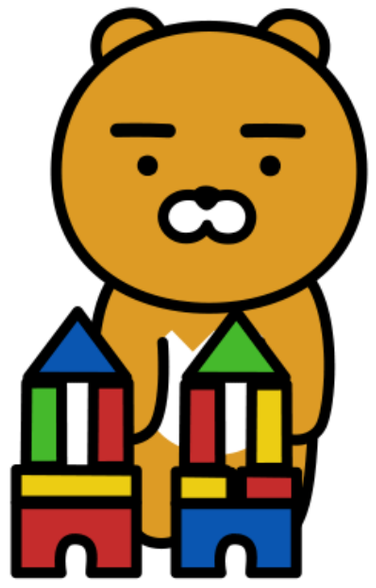
플랫폼의 결정

- JVM - Java - Spring Boot
- Message Queue - Kafka (Pub/Sub)
- ~~Spring Cloud Stream~~
- 그런데 여기서 잠깐만요.



플랫폼의 결정

- JVM - Java - Spring Boot
- Message Queue - Kafka (Pub/Sub)
- Message Queue Client Library
- Spring Cloud Stream와 비교하면?



Spring Cloud Stream과 비교

Spring Cloud Stream과 비교

- 좋은 점은?

Spring Cloud Stream과 비교

- 좋은 점은? - 비즈니스 로직과 설정의 분리

Spring Cloud Stream과 비교

- 좋은 점은? - 비즈니스 로직과 설정의 분리
- 기존에는

Spring Cloud Stream과 비교

- 좋은 점은? - 비즈니스 로직과 설정의 분리
- 기존에는 - 설정이 로직 안으로 들어감

```
Properties properties = new Properties();
properties.put("zookeeper.connect", “zookeeper server address");
properties.put("bootstrap.servers", “kafka server address");
properties.put("acks", "1");
properties.put("retries", 0);
properties.put("linger.ms", 1);
properties.put("key.serializer", "org.apache.kafka.common.serialization.StringSerializer");
properties.put("value.serializer", "channel.MessageInterfaceSerializer");
```

Spring Cloud Stream과 비교

- 좋은 점은? - 비즈니스 로직과 설정의 분리
- Spring Cloud Stream은

```
spring:
  cloud:
    stream:
      kafka:
        binder:
          autoCreateTopics: false
          autoAddPartitions: false
          zkNodes: Zookeeper Server Address
          brokers: Kafka Server Address
      ....
```

Spring Cloud Stream과 비교

- 좋은 점은? - 비즈니스 로직과 설정의 분리
- Spring Cloud Stream은 - 설정 파일로 분리

```
spring:
  cloud:
    stream:
      kafka:
        binder:
          autoCreateTopics: false
          autoAddPartitions: false
          zkNodes: Zookeeper Server Address
          brokers: Kafka Server Address
      ....
```

Spring Cloud Stream과 비교

- 좋은 점은? - 비즈니스 로직과 설정의 분리
- Spring Cloud Stream은 - 설정 파일로 분리
- YML, Properties의 자동완성기능 사용가능

```
spring:
  cloud:
    stream:
      kafka:
        binder:
          autoCreateTopics: false
          autoAddPartitions: false
          zkNodes: Zookeeper Server Address
          brokers: Kafka Server Address
      ....
```

Spring Cloud Stream과 비교

- 좋은 점은? - 비즈니스 로직과 설정의 분리
- Spring Cloud Stream은 - 설정 파일로 분리
- YML, Properties의 자동완성기능 사용가능

```
spring:
  cloud:
    stream:
      kafka:
        binder:
          autoCreateTopics: false
          autoAddPartitions: false
          zkNodes: Zookeeper Server Address
          brokers: Kafka Server Address
    ....
```

- local
- dev
- stage
- real

Spring Cloud Stream과 비교

- 또 좋은 점은? - Code량 감소

Spring Cloud Stream과 비교

- 또 좋은 점은? - Code량 감소
- Serializer, Deserializer 구현 필요 없음

Spring Cloud Stream과 비교

- 또 좋은 점은? - Code량 감소
- Serializer, Deserializer 구현 필요 없음
- 개발자는 비즈니스 로직에 집중 가능

좀더 자세한 Code는 Demo에서

플랫폼의 결정

- JVM - Java - Spring Boot
- Message Queue - Kafka (Pub/Sub)
- Spring Cloud Stream

Spring Cloud Stream

Spring Cloud Stream

- Spring Cloud

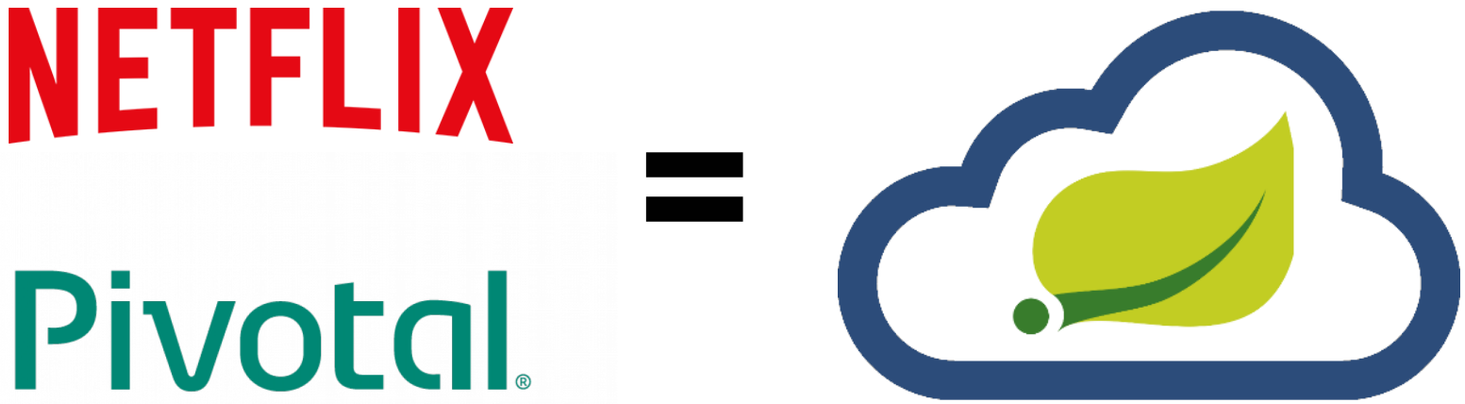
Spring Cloud Stream

- Spring Cloud
- Stream

Spring Cloud란 무엇일까요?

Spring Cloud란 무엇일까요?

- Pivotal과 Netflix의 합작품



Spring Cloud란 무엇일까요?

- Pivotal과 Netflix의 합작품
- Netflix OSS

NETFLIX

Pivotal

=

- Service Discovery
- Hystrix
- Zuul
- Ribbon

Spring Cloud란 무엇일까요?

- Pivotal과 Netflix의 합작품
- Netflix OSS와 통합

NETFLIX

Pivotal

=

- Service Discovery
- Hystrix
- Zuul
- Ribbon

Spring Cloud란 무엇일까요?

- Pivotal과 Netflix의 합작품
- Netflix OSS와 통합
- Spring Boot위에서 동작

NETFLIX

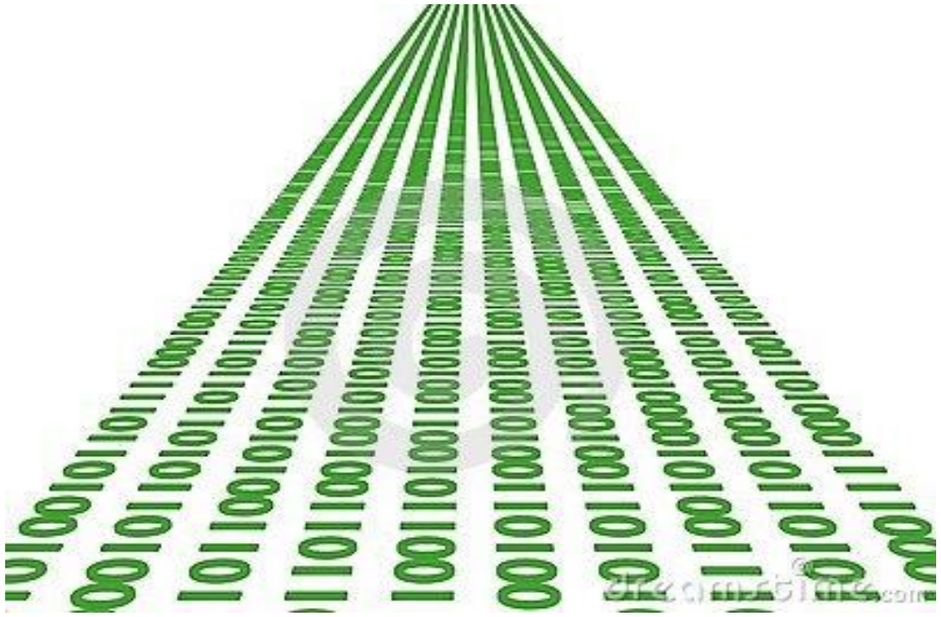
Pivotal

=

Stream이란 무엇일까요?

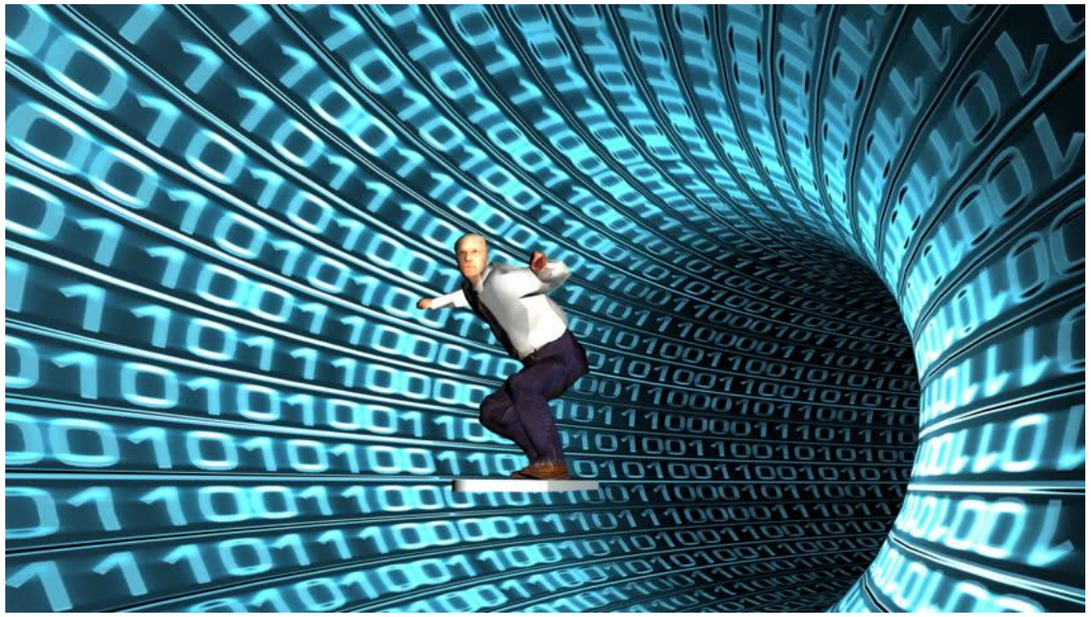
Stream이란 무엇일까요?

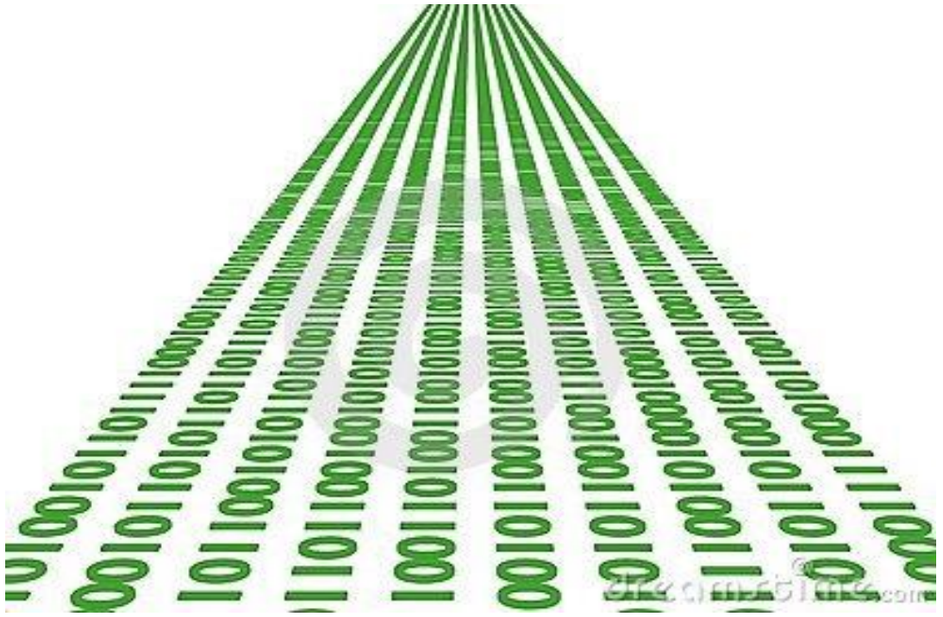
- 데이터의 흐름



Stream이란 무엇일까요?

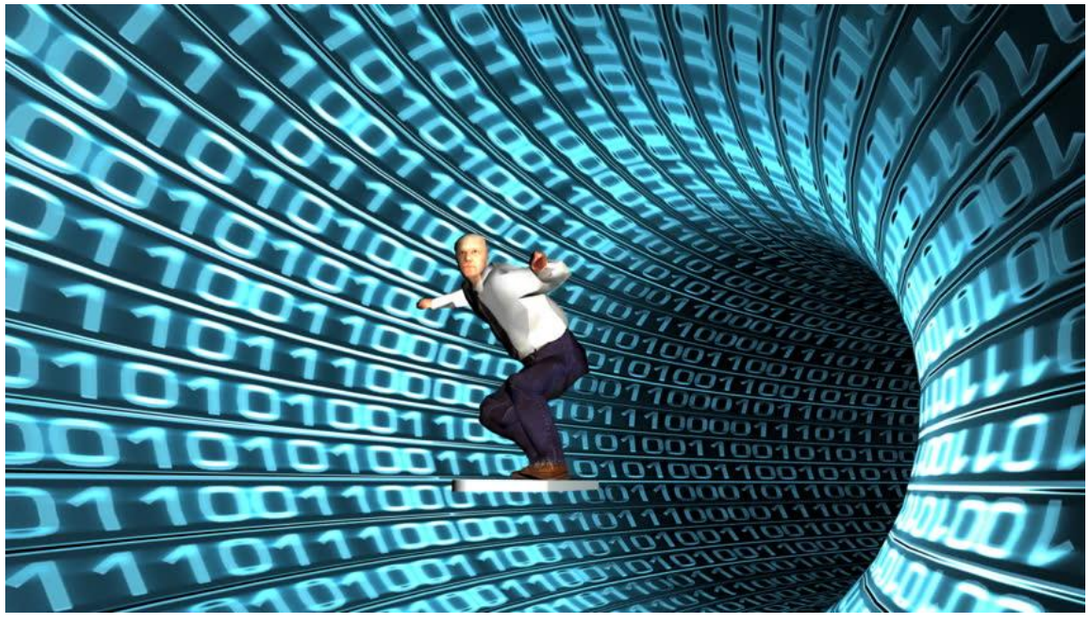
- 데이터의 흐름
- Queue를 이용한 가상의 채널

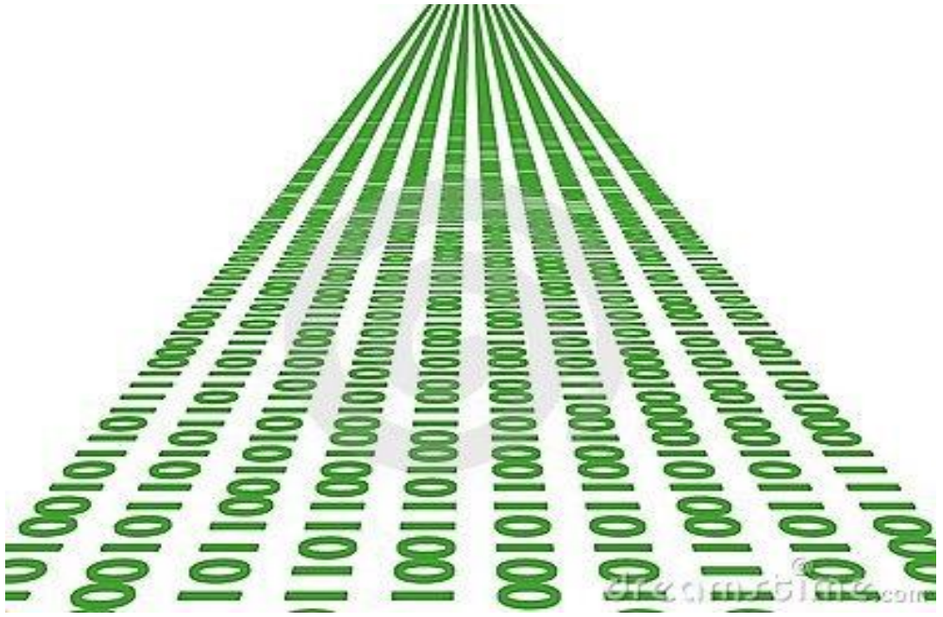




Stream이란 무엇일까요?

- 데이터의 흐름
- Queue를 이용한 가상의 채널
- 실시간으로 데이터를 처리하기 적합





데이터를 처리하는 방식

데이터를 처리하는 방식

- Batch

데이터를 처리하는 방식

- Batch (Micro Batch)

데이터를 처리하는 방식

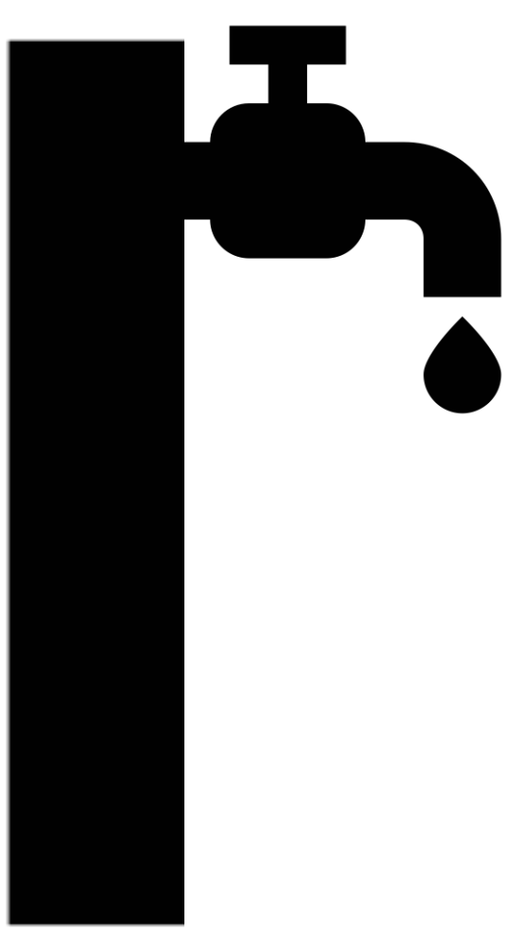
- Batch (Micro Batch)
- Stream

데이터를 처리하는 방식

- Batch (Micro Batch)
- Stream

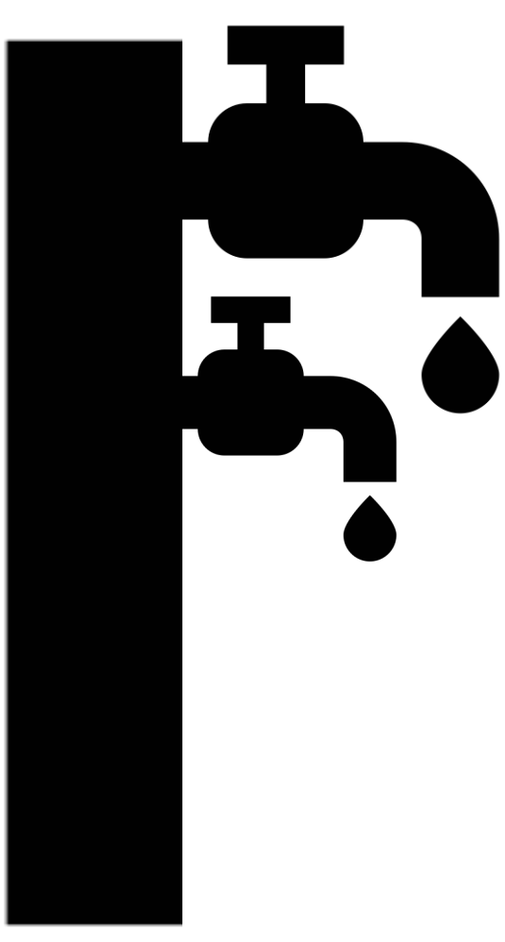
데이터를 처리하는 방식

- Batch (Micro Batch)
- Stream



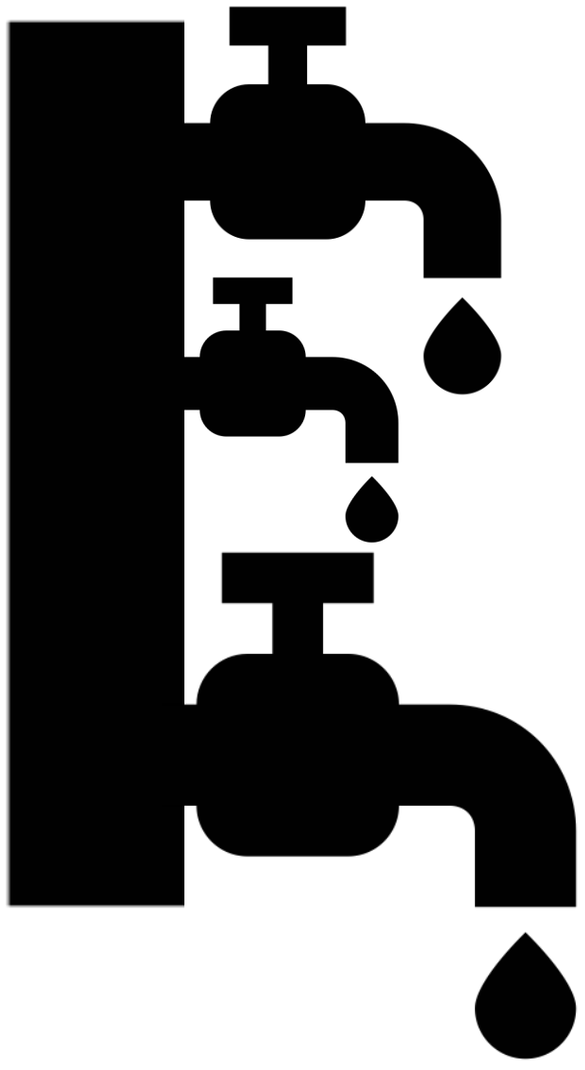
데이터를 처리하는 방식

- Batch (Micro Batch)
- Stream



데이터를 처리하는 방식

- Batch (Micro Batch)
- Stream



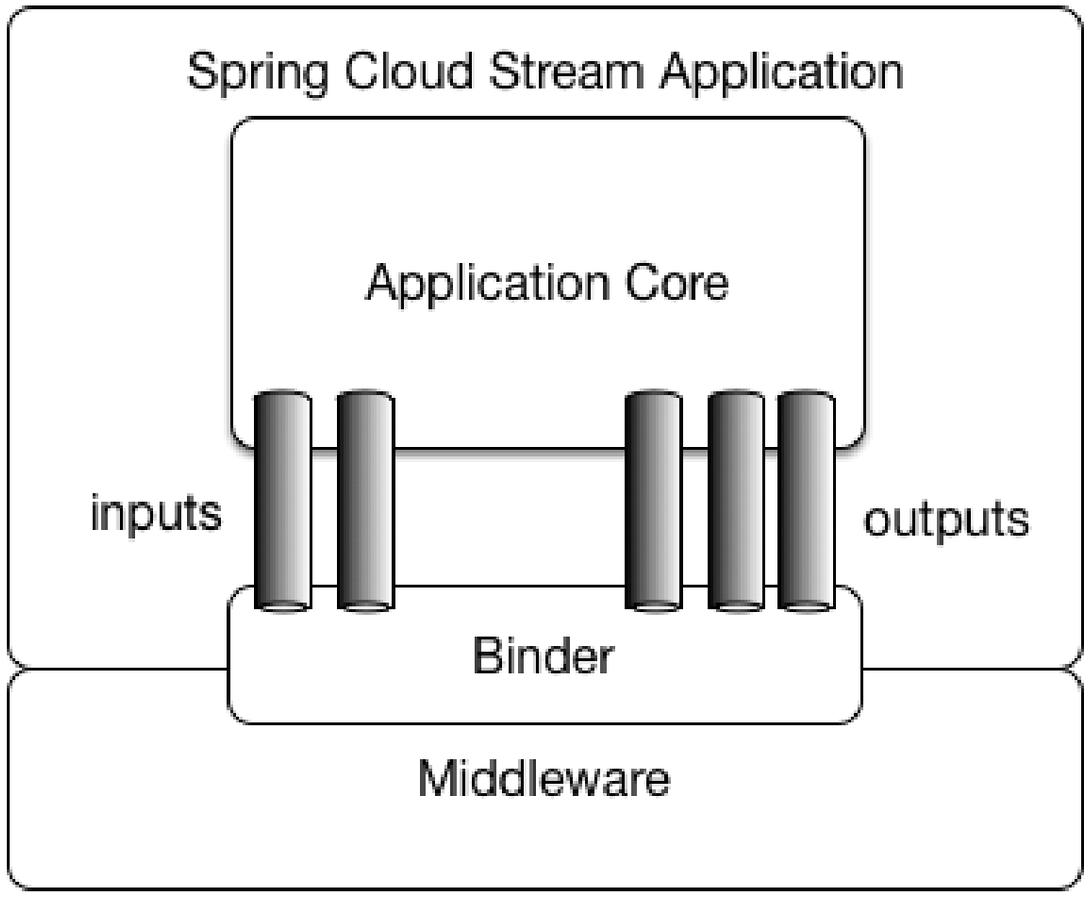
Spring Cloud Stream이란 무엇일까요?

Spring Cloud Stream이란 무엇일까요?

- Event,Message-Driven Microservice

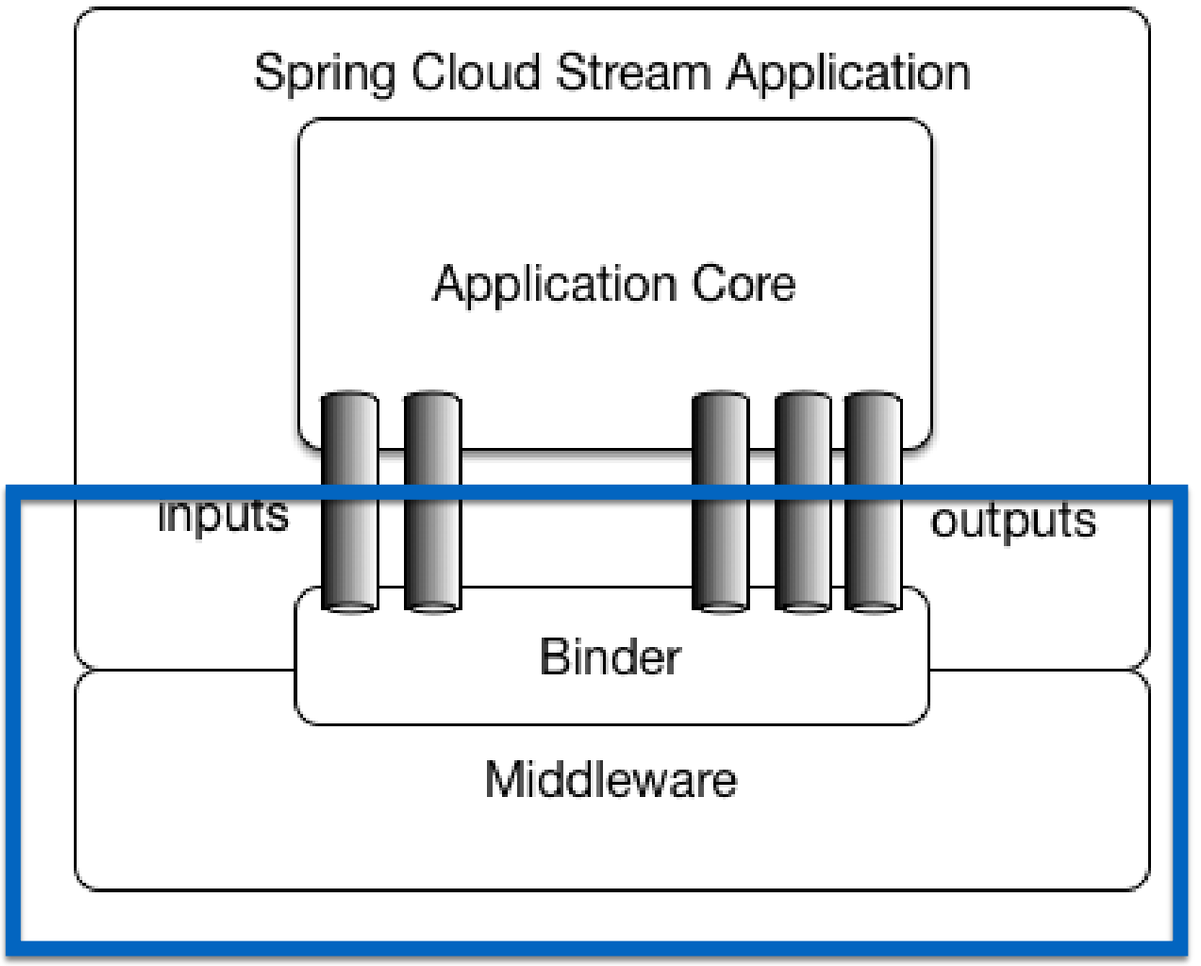
Spring Cloud Stream이란 무엇일까요?

- Event,Message-Driven Microservice



Spring Cloud Stream이란 무엇일까요?

- Event,Message-Driven Microservice
- Middleware - Message Queue

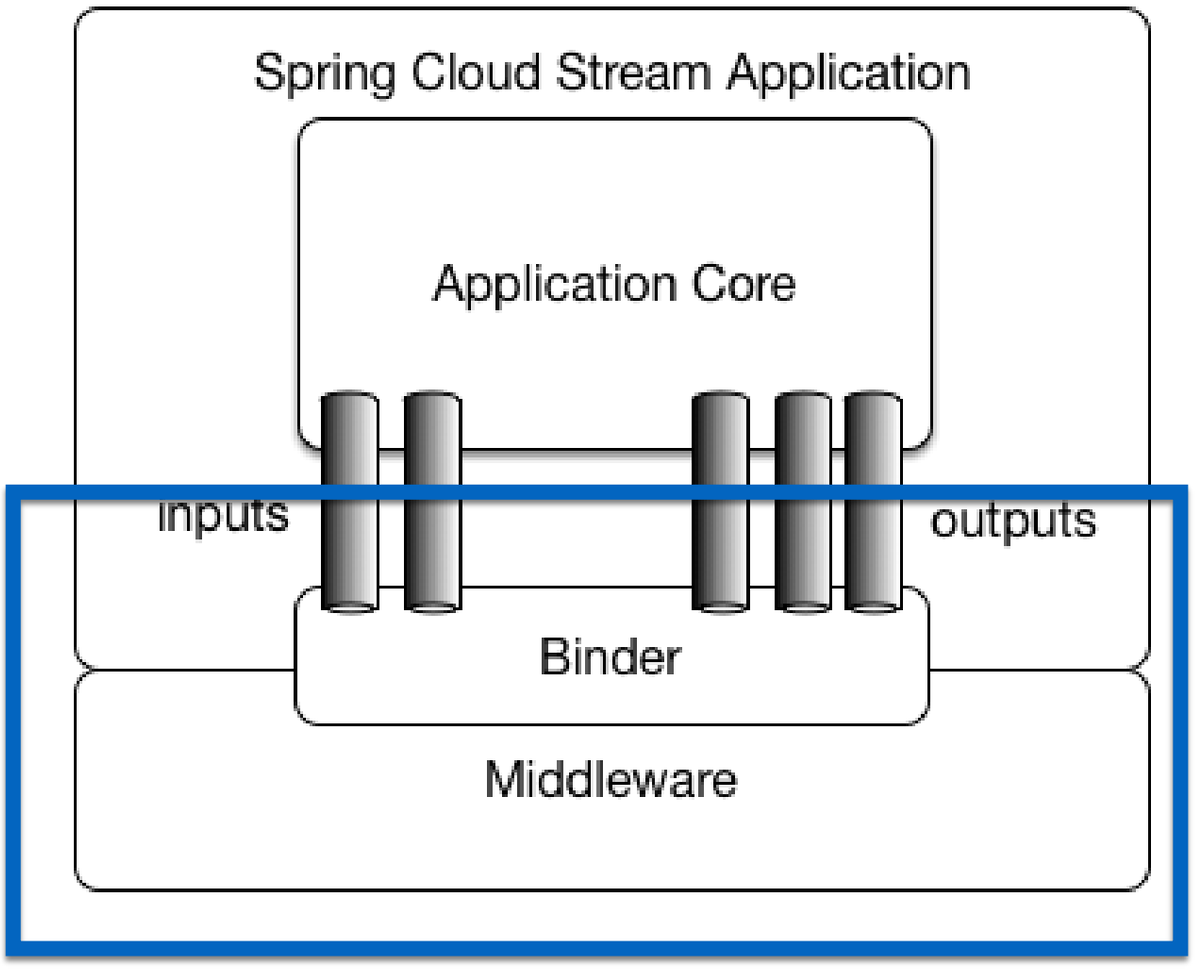


Spring Cloud Stream이란 무엇일까요?

- Event,Message-Driven Microservice
- Middleware - Message Queue

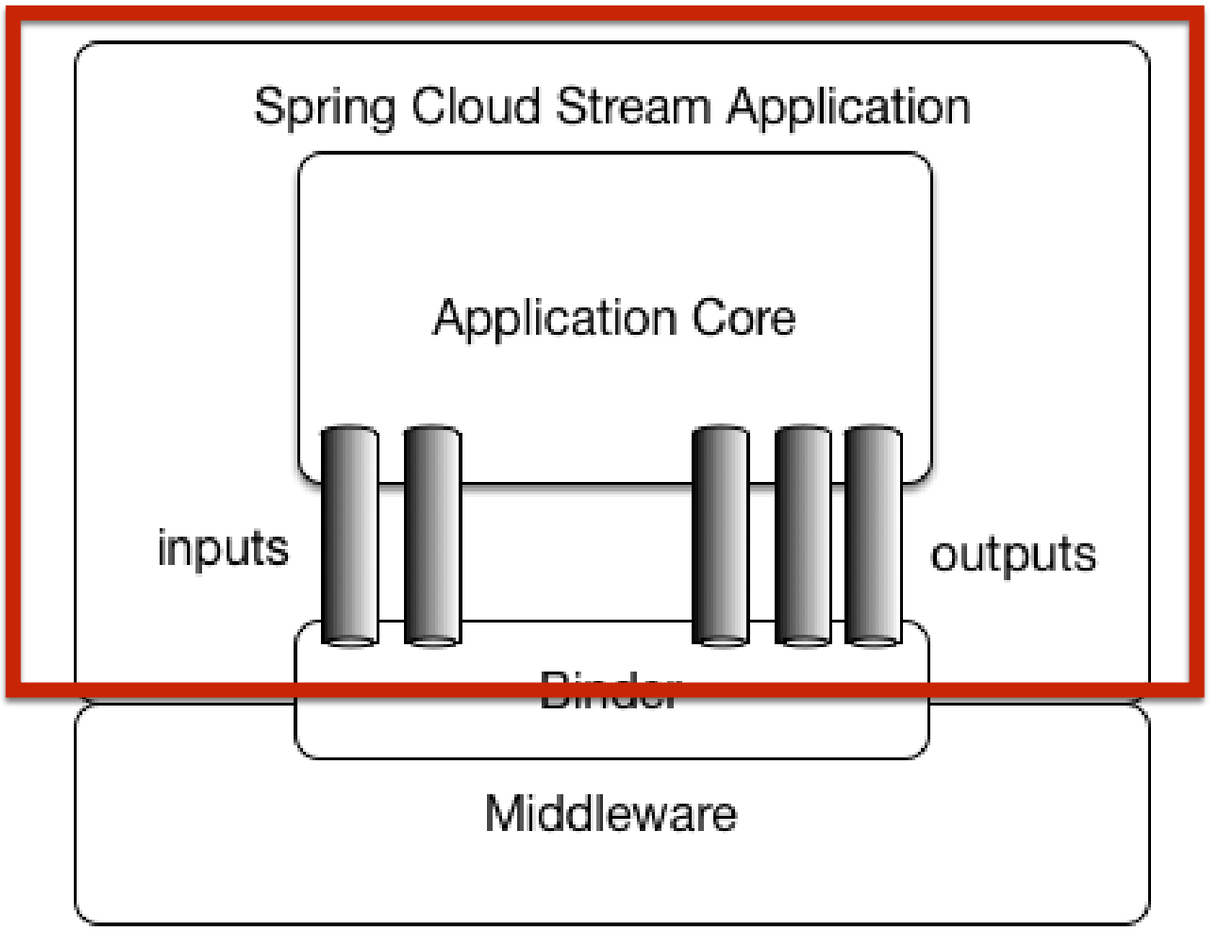
APACHE kafka™
A distributed streaming platform

RabbitMQ



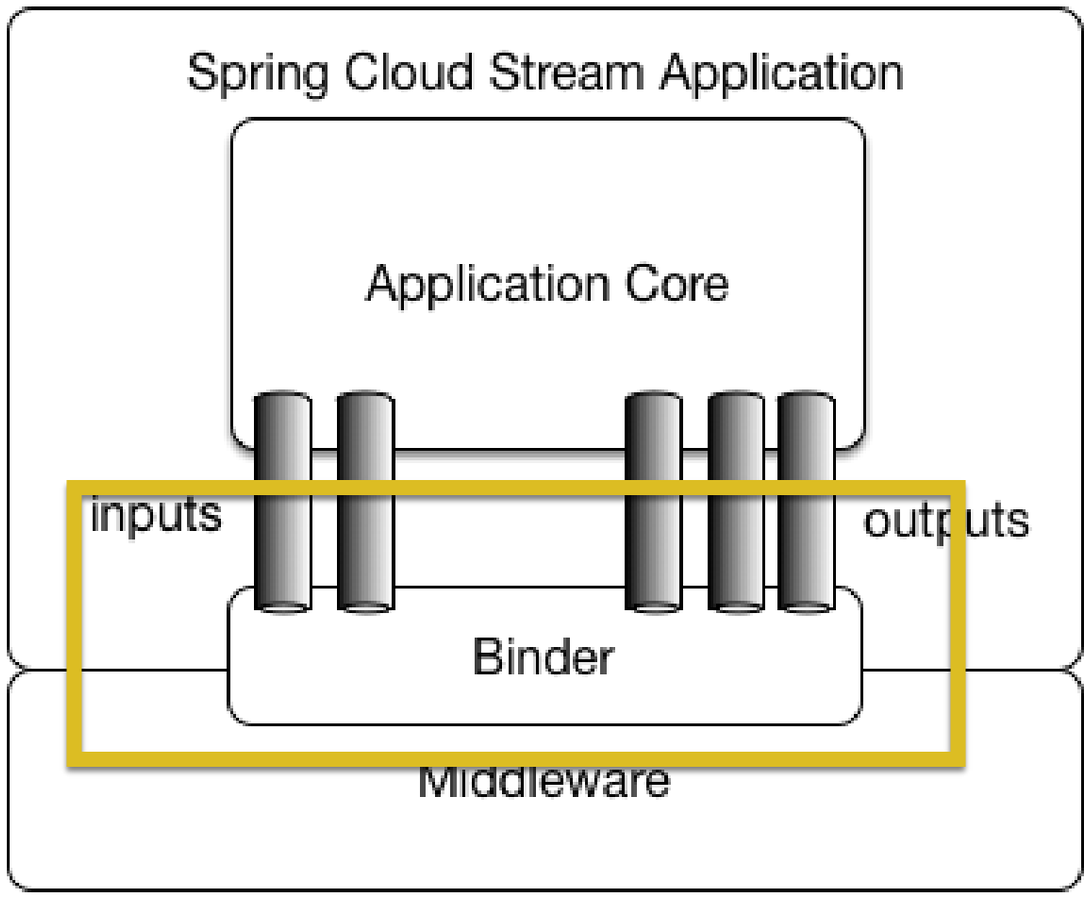
Spring Cloud Stream이란 무엇일까요?

- Spring Cloud Stream Application



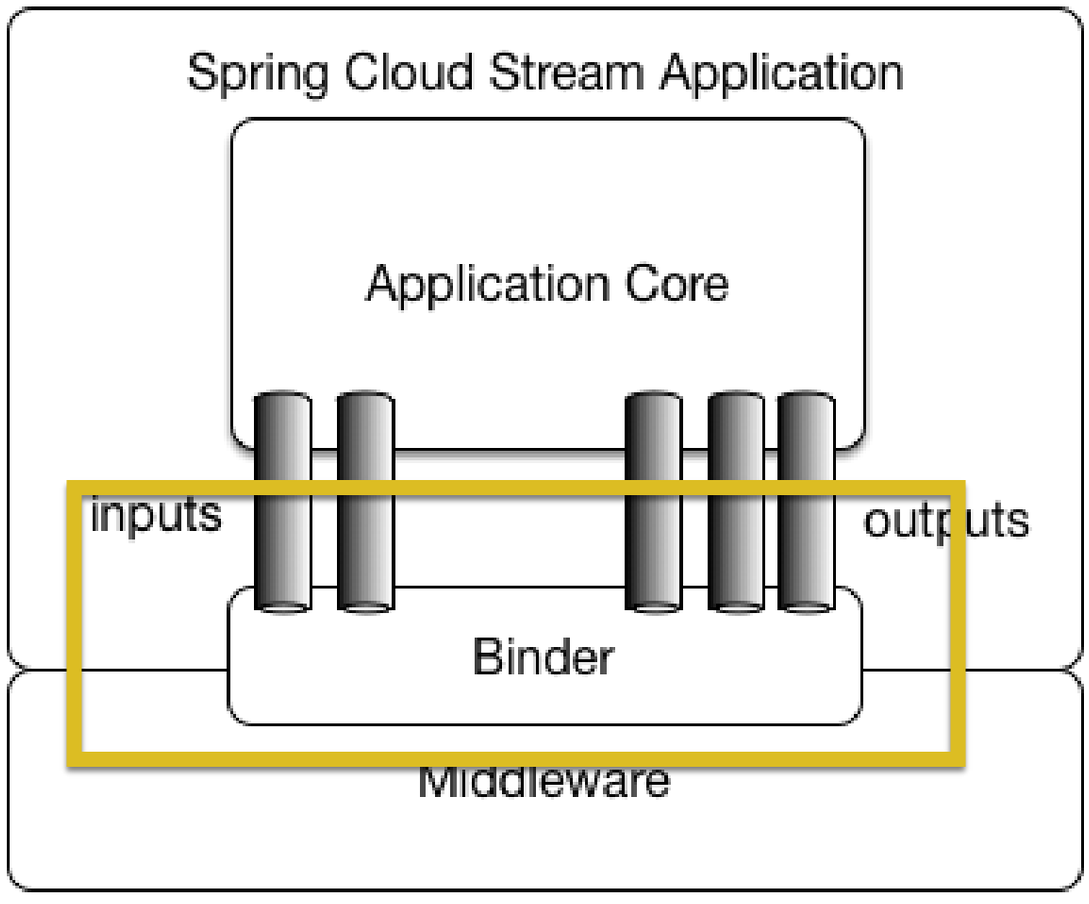
Spring Cloud Stream이란 무엇일까요?

- Binder



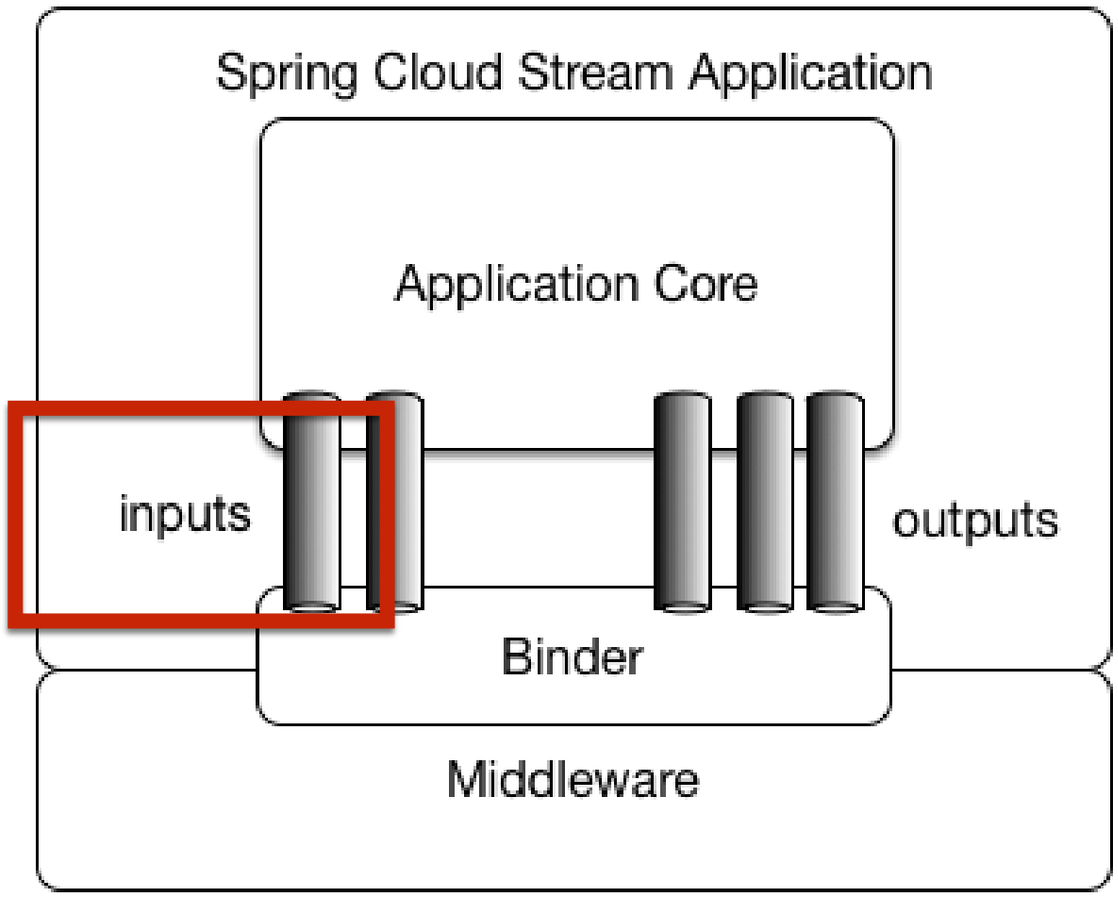
Spring Cloud Stream이란 무엇일까요?

- Binder
- Application ⟷ Message Queue



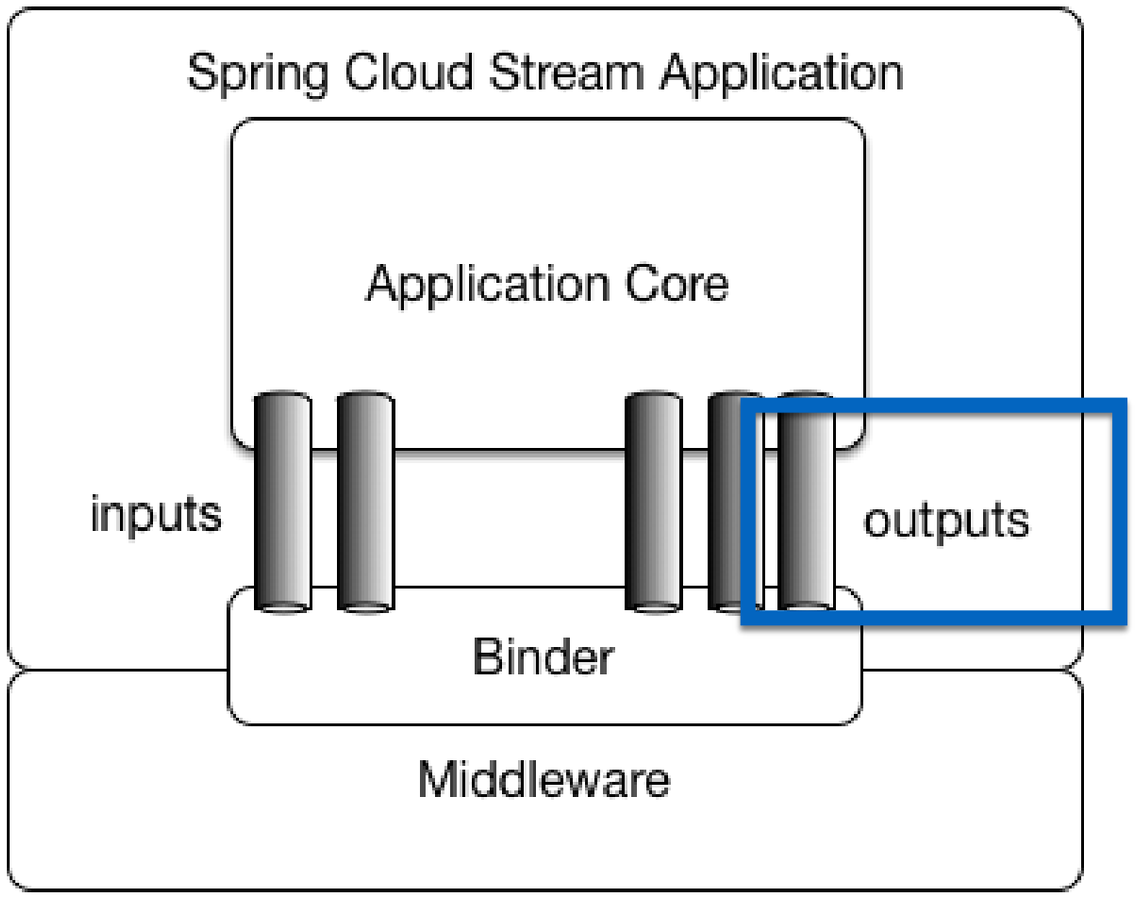
Spring Cloud Stream이란 무엇일까요?

- Input Channel - **Subscribe**



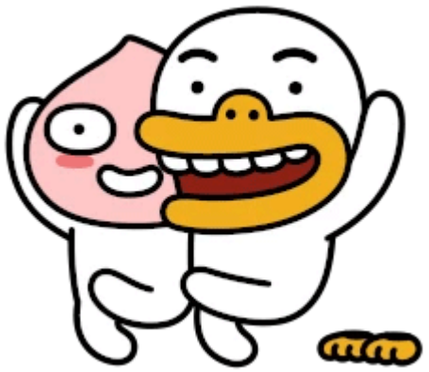
Spring Cloud Stream이란 무엇일까요?

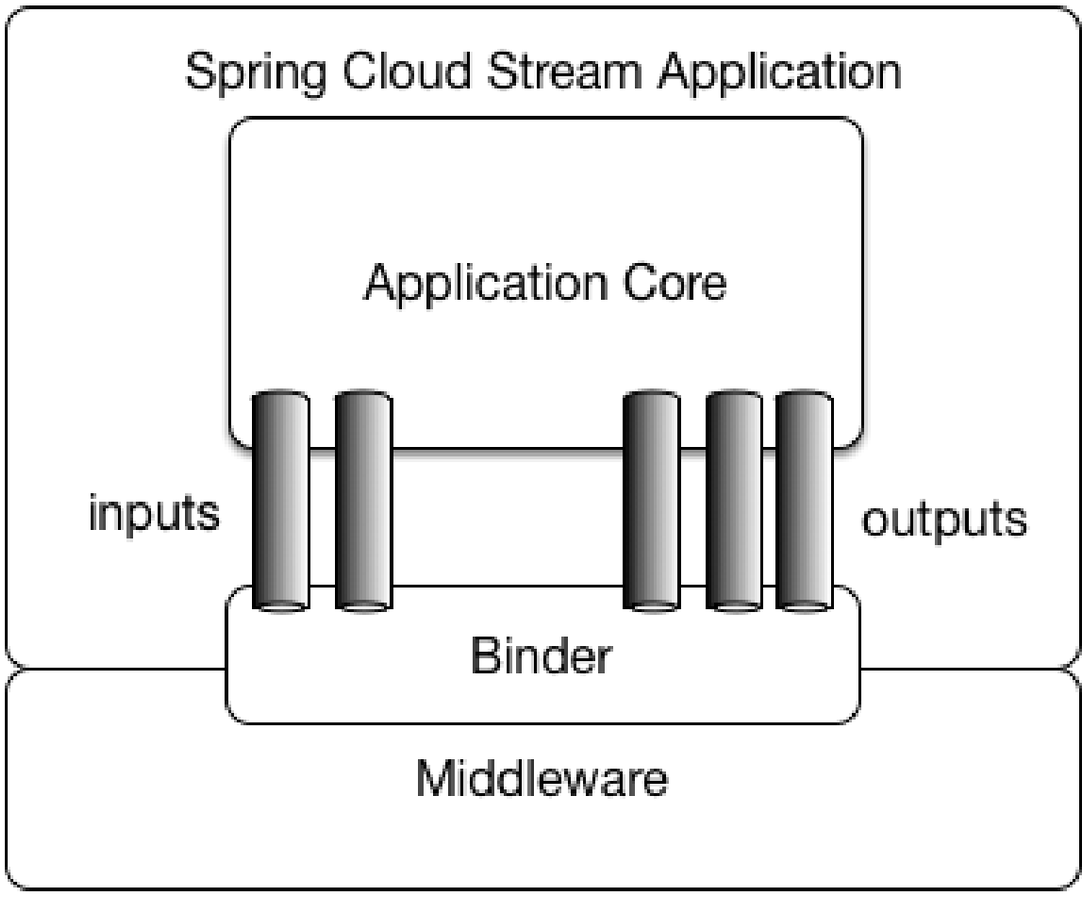
- Input Channel - **Subscribe**
- Output Channel - **Publish**



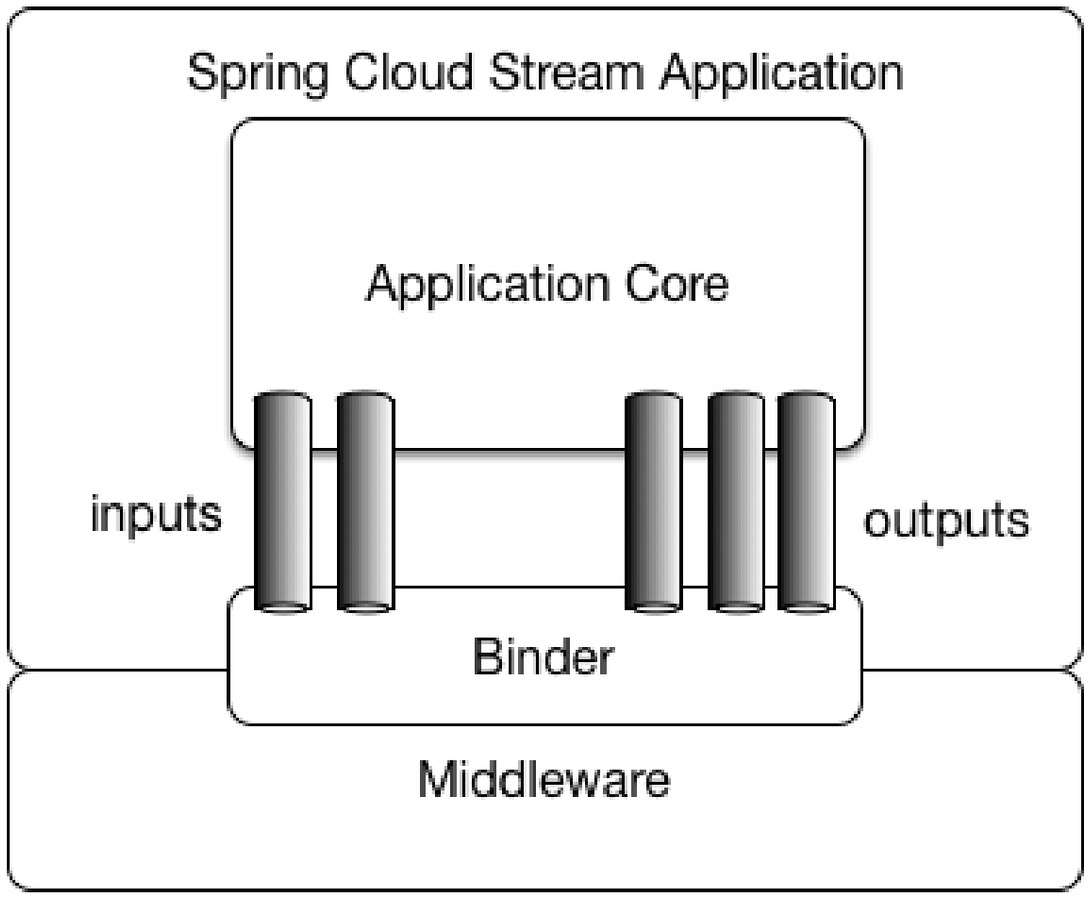
Spring Cloud Stream이란 무엇일까요?

- **Easy to Build by Annotation**



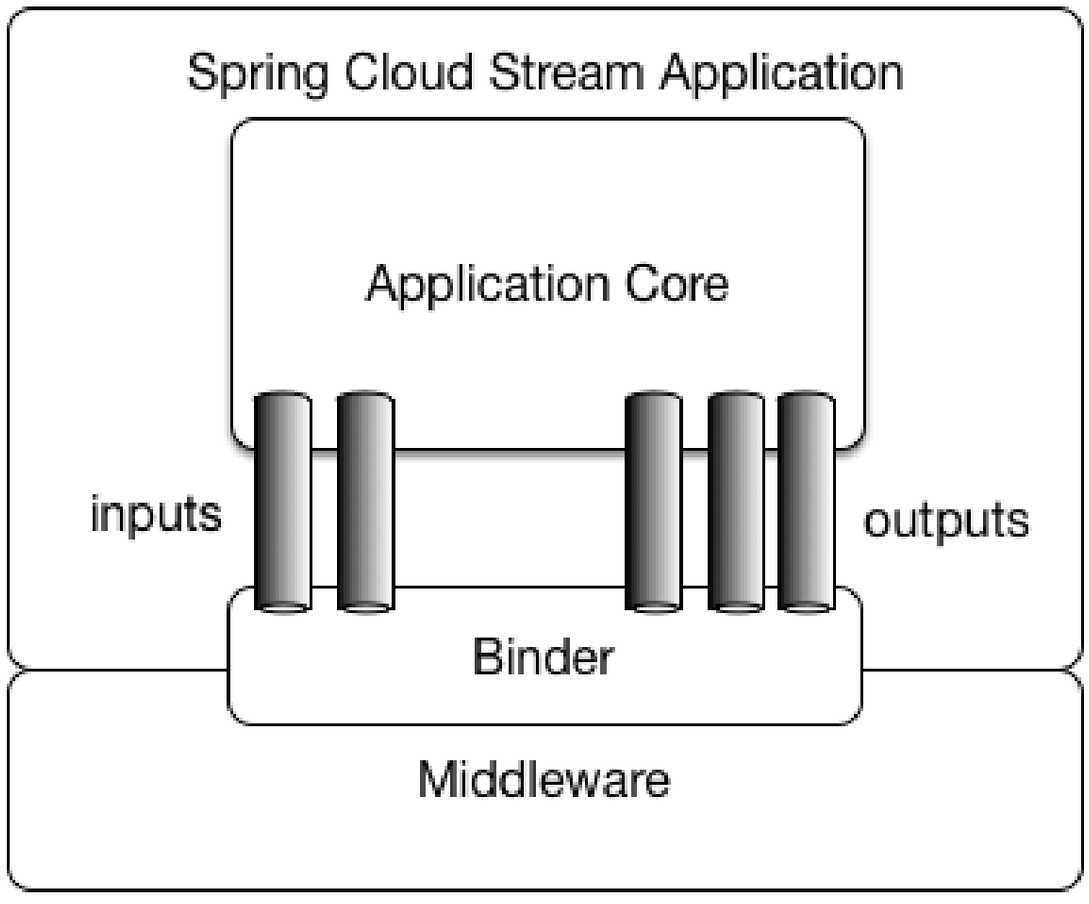


Spring Cloud Stream의 에너테이션



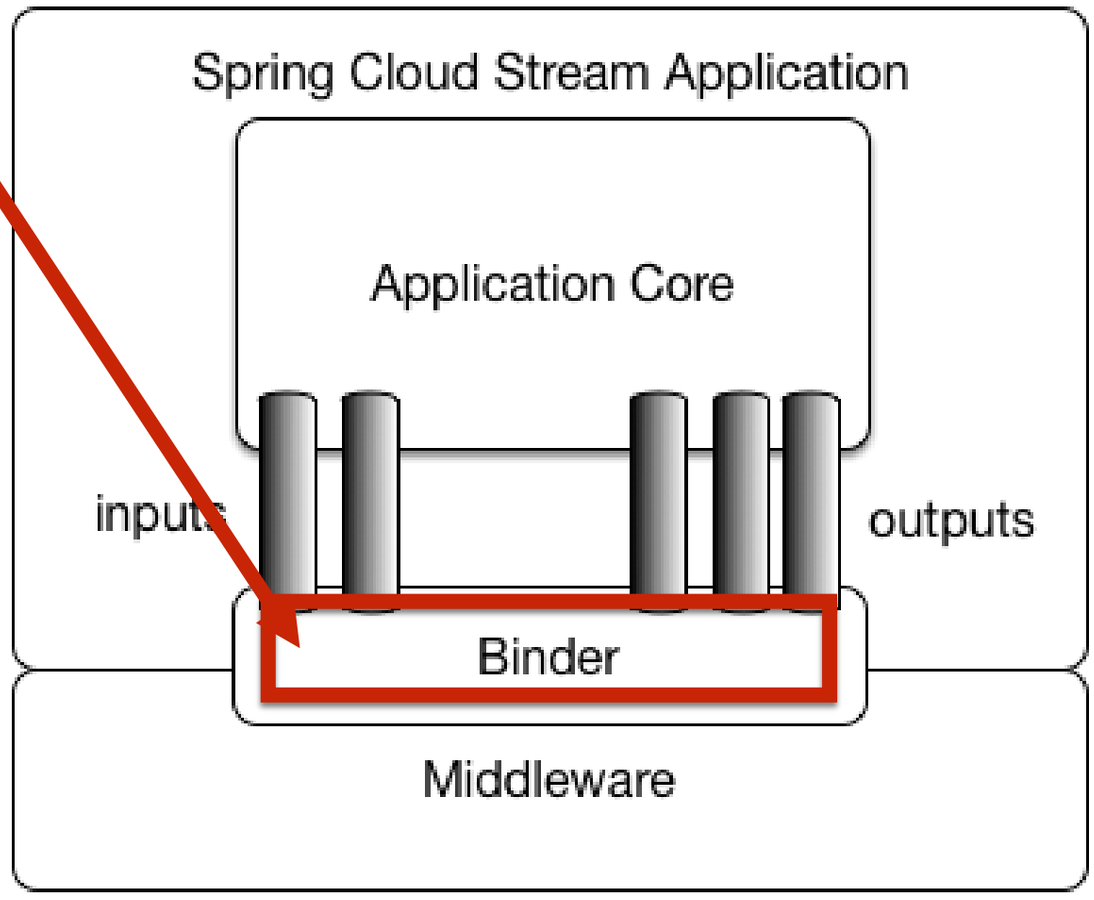
Spring Cloud Stream의 에너테이션

- @EnableBinding



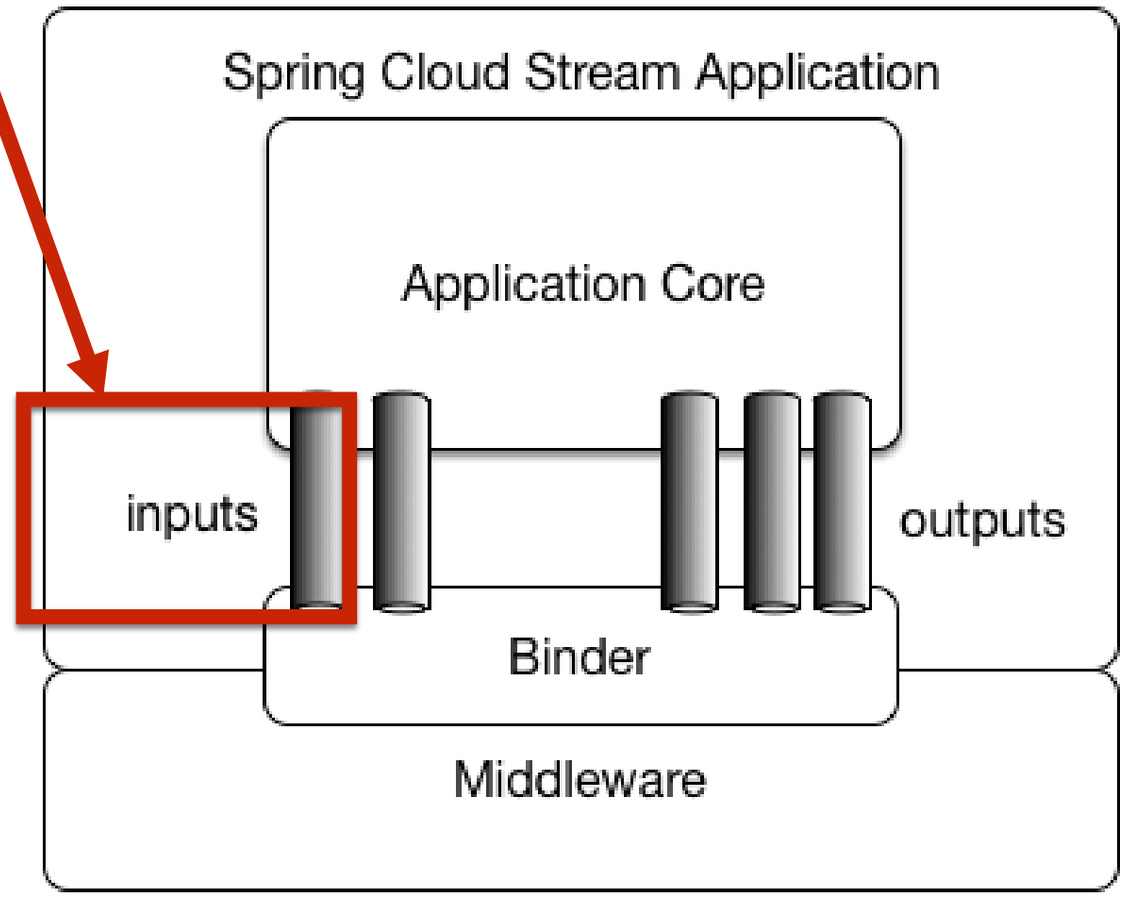
Spring Cloud Stream의 에너테이션

- @EnableBinding



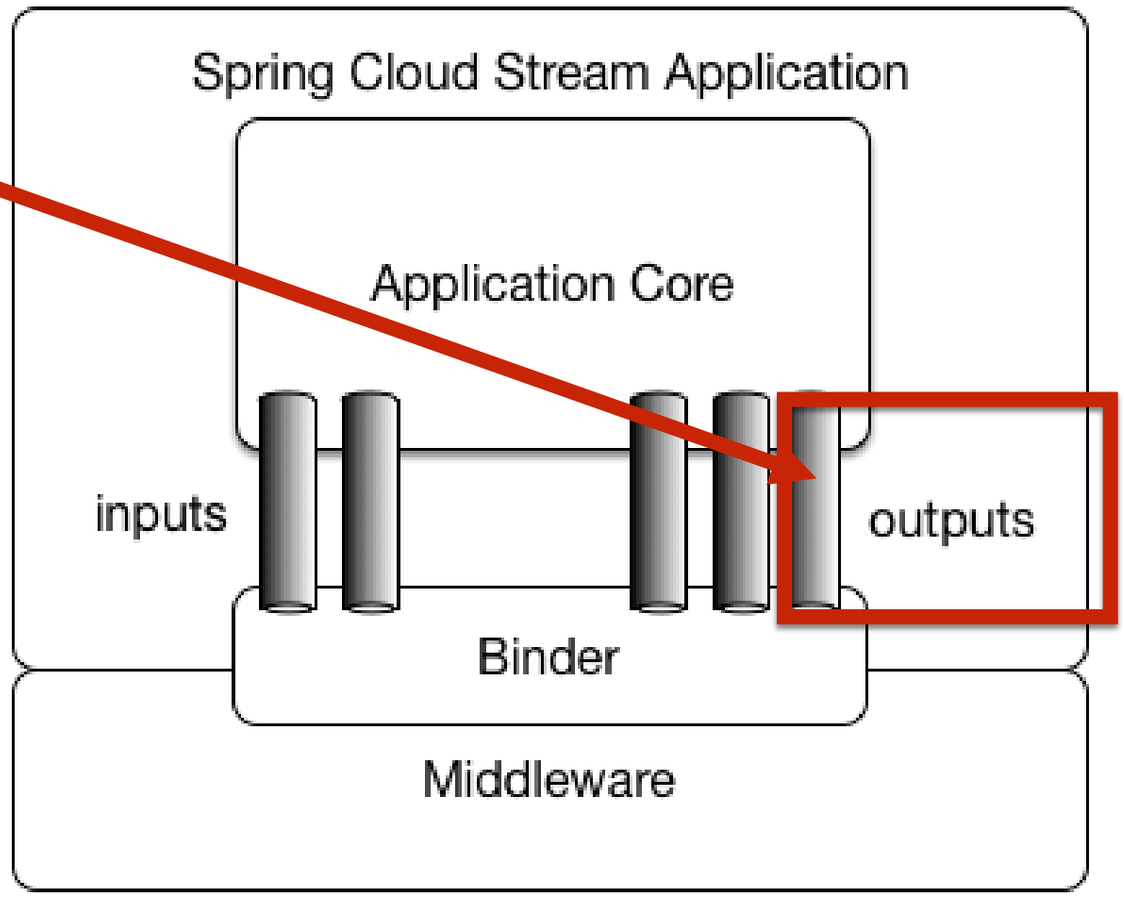
Spring Cloud Stream의 에너테이션

- @EnableBinding
  - Sink interface



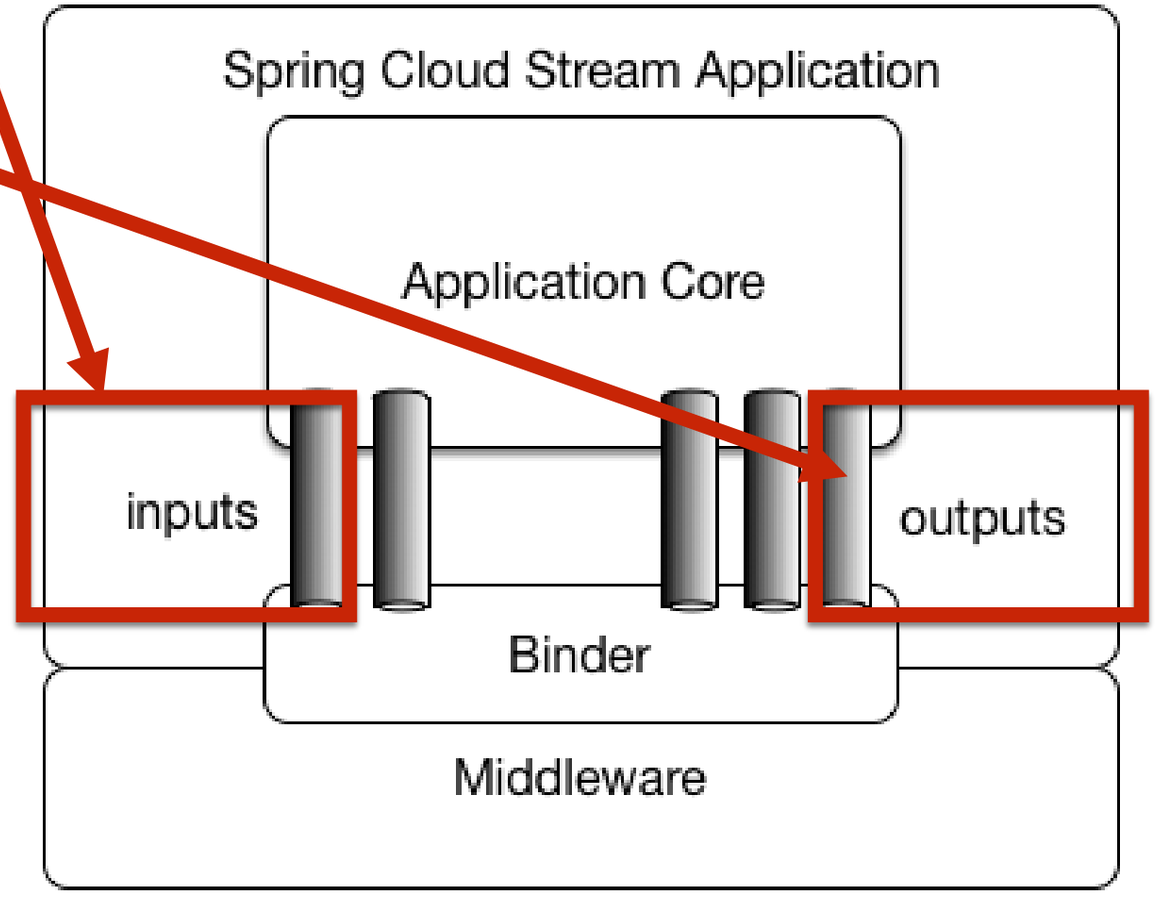
Spring Cloud Stream의 에너테이션

- @EnableBinding
  - Sink interface
  - Source interface



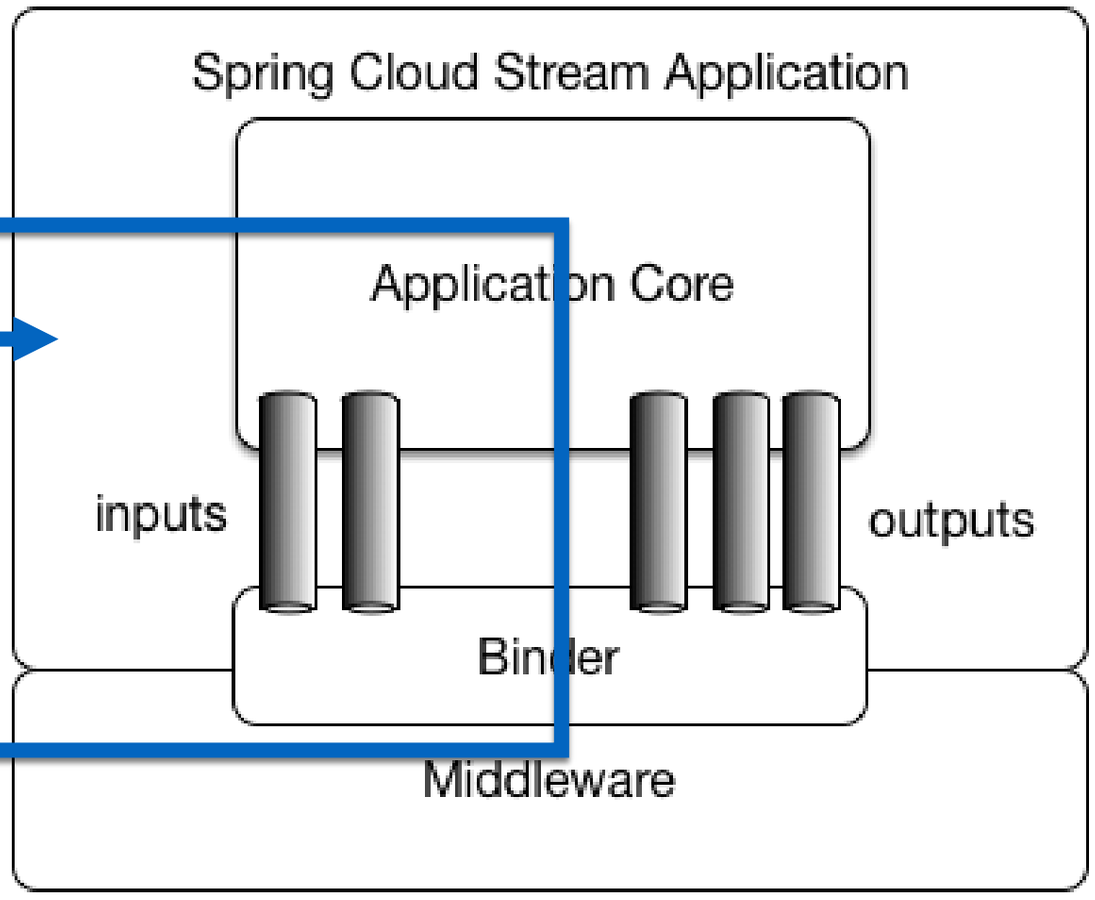
Spring Cloud Stream의 에너테이션

- @EnableBinding
  - Sink interface
  - Source interface



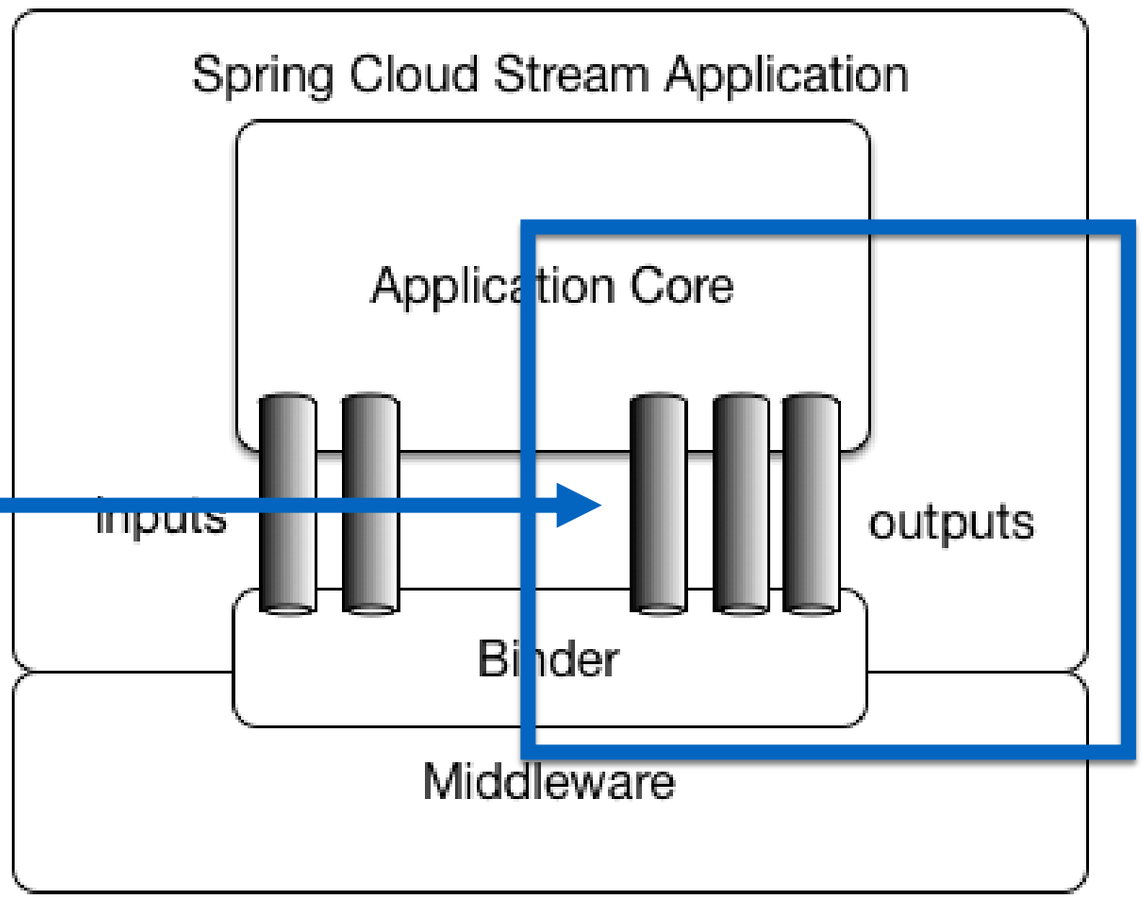
Spring Cloud Stream의 에너테이션

- @EnableBinding
  - Sink interface
  - Source interface
- @StreamListener



Spring Cloud Stream의 에너테이션

- @EnableBinding
  - Sink interface
  - Source interface
- @StreamListener
- @InboundChannelAdapter



Message Queue는 어떤것을 사용할까?

Message Queue는 어떤것을 사용할까?

Message Queue는 어떤것을 사용할까?

Message Queue는 어떤것을 사용할까?

Message Queue는 어떤것을 사용할까?

- FileSystem에 저장

- Memory에 저장

Message Queue는 어떤것을 사용할까?

- FileSystem에 저장
- TCP/IP

- Memory에 저장
- AMQP

Message Queue는 어떤것을 사용할까?

- FileSystem에 저장
- TCP/IP
- 뛰어난 성능

- Memory에 저장
- AMQP
- 다양한 기능

Message Queue는 어떤것을 사용할까?

- FileSystem에 저장
- TCP/IP
- 뛰어난 성능?

- Memory에 저장
- AMQP
- 다양한 기능

Message Queue는 어떤것을 사용할까?

- FileSystem에 저장
- TCP/IP
- 뛰어난 성능?

- Memory에 저장
- AMQP
- 다양한 기능

Message Queue는 어떤것을 사용할까?

APACHE kafka™
A distributed streaming platform

RabbitMQ

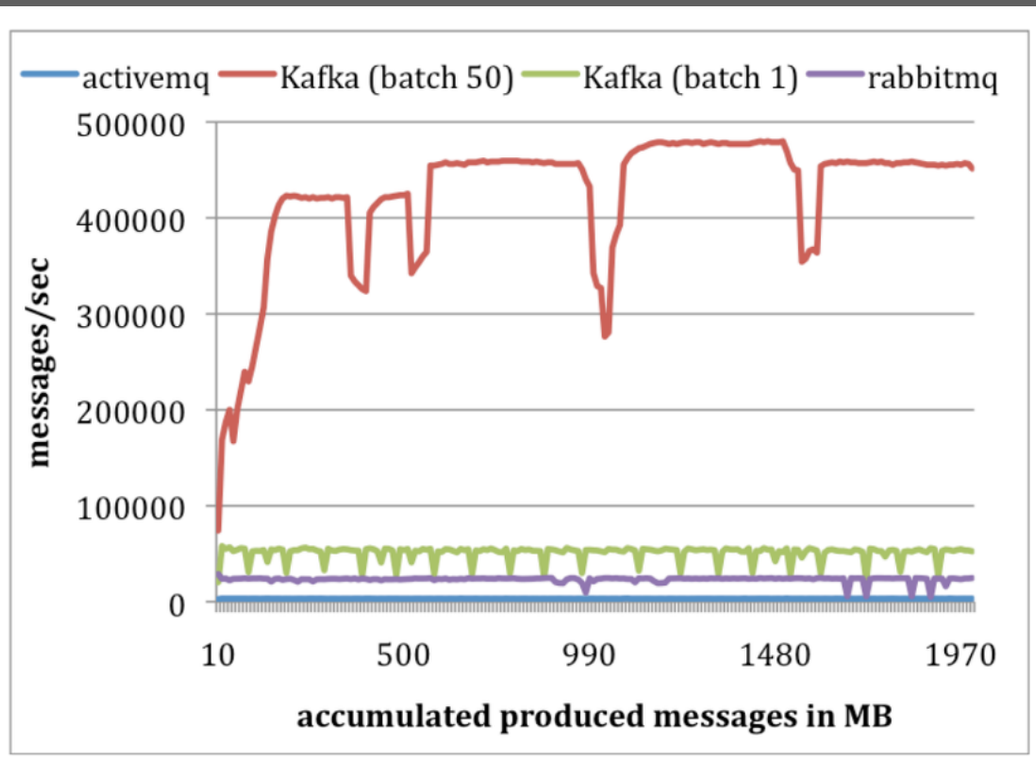


Figure 4. Producer Performance

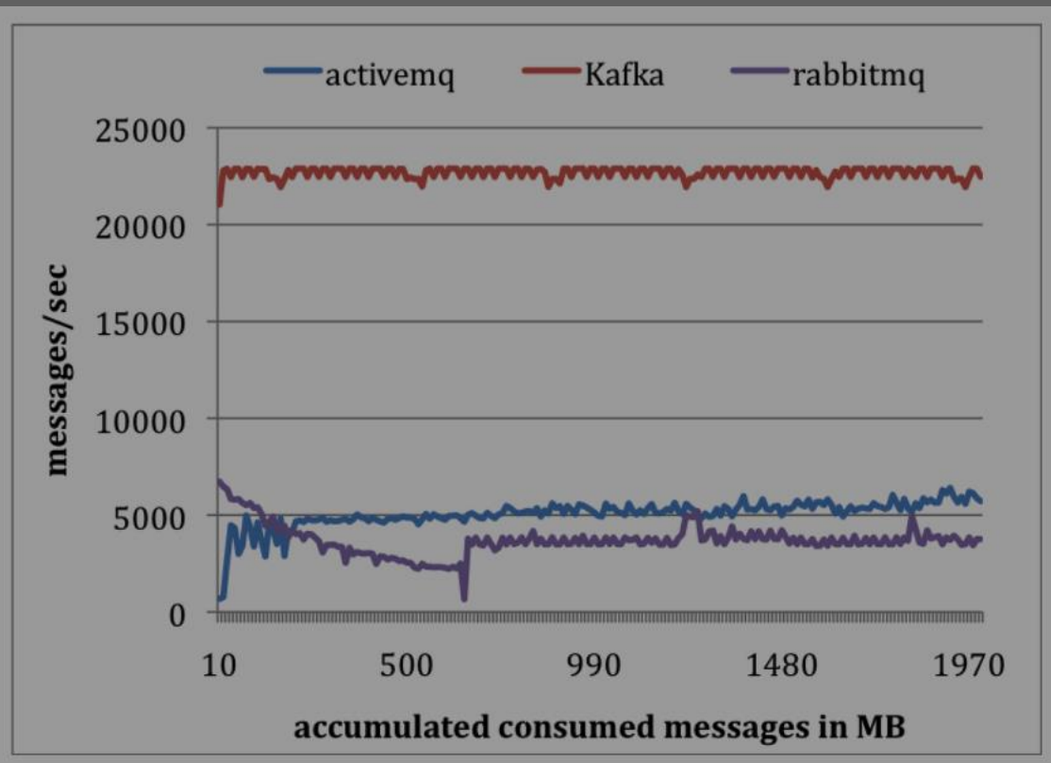


Figure 5. Consumer Performance

kakao

Message Queue는 어떤것을 사용할까?

APACHE kafka™
A distributed streaming platform

RabbitMQ

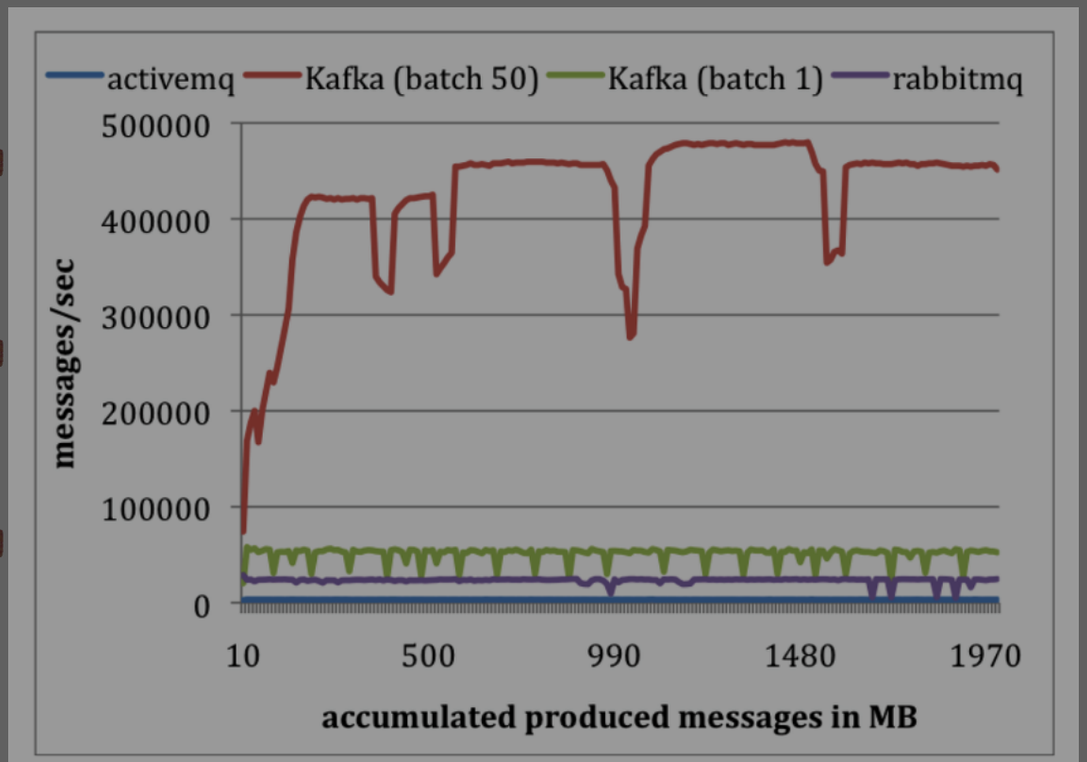


Figure 4. Producer Performance

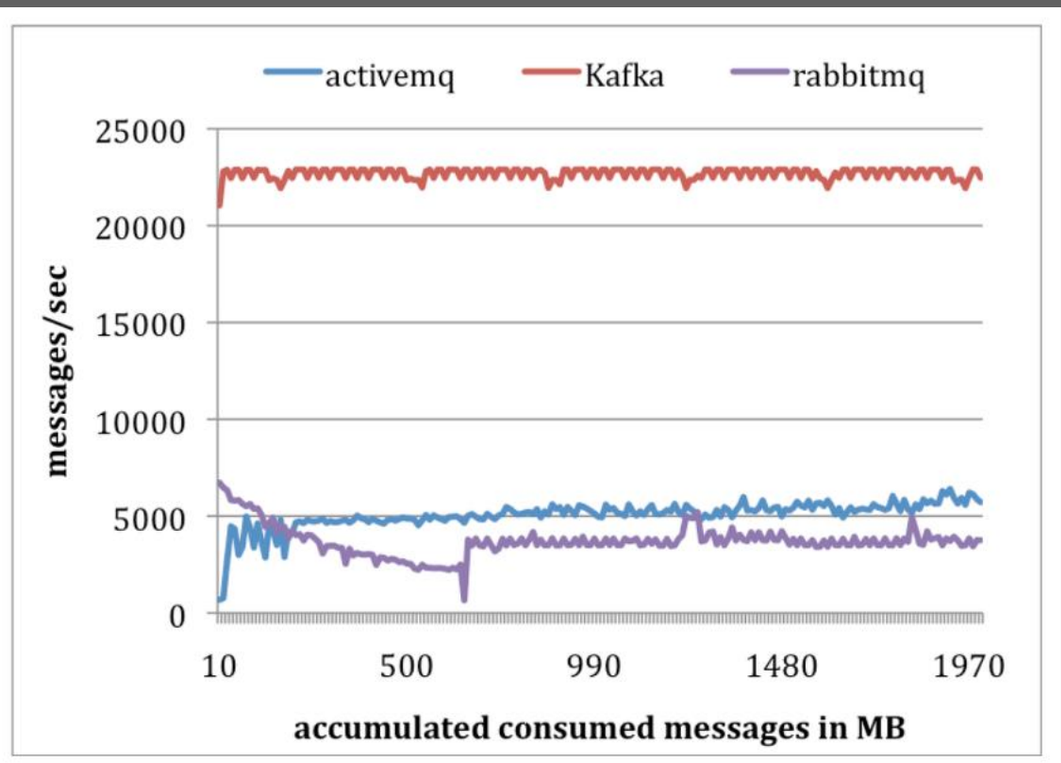
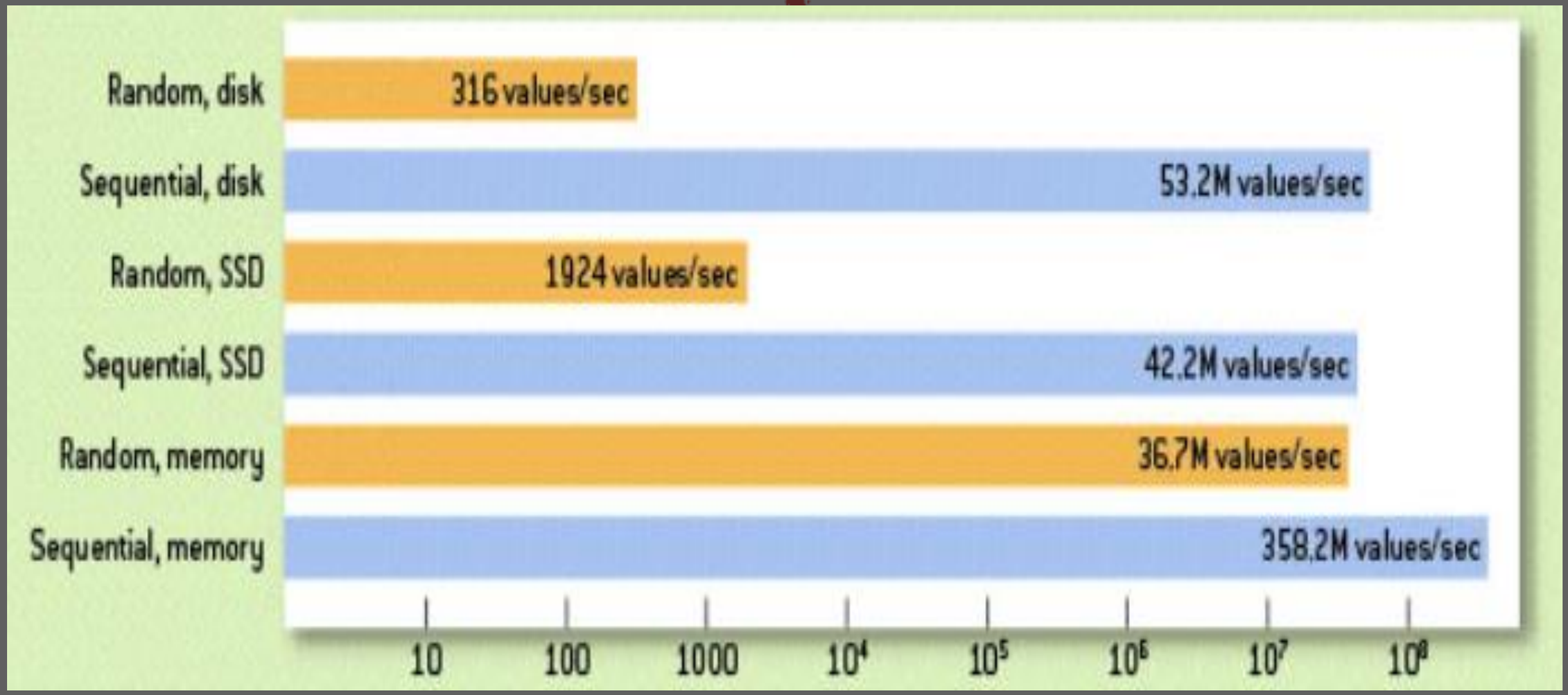


Figure 5. Consumer Performance

Message Queue는 어떤것을 사용할까?

APACHE kafka™
A distributed streaming platform

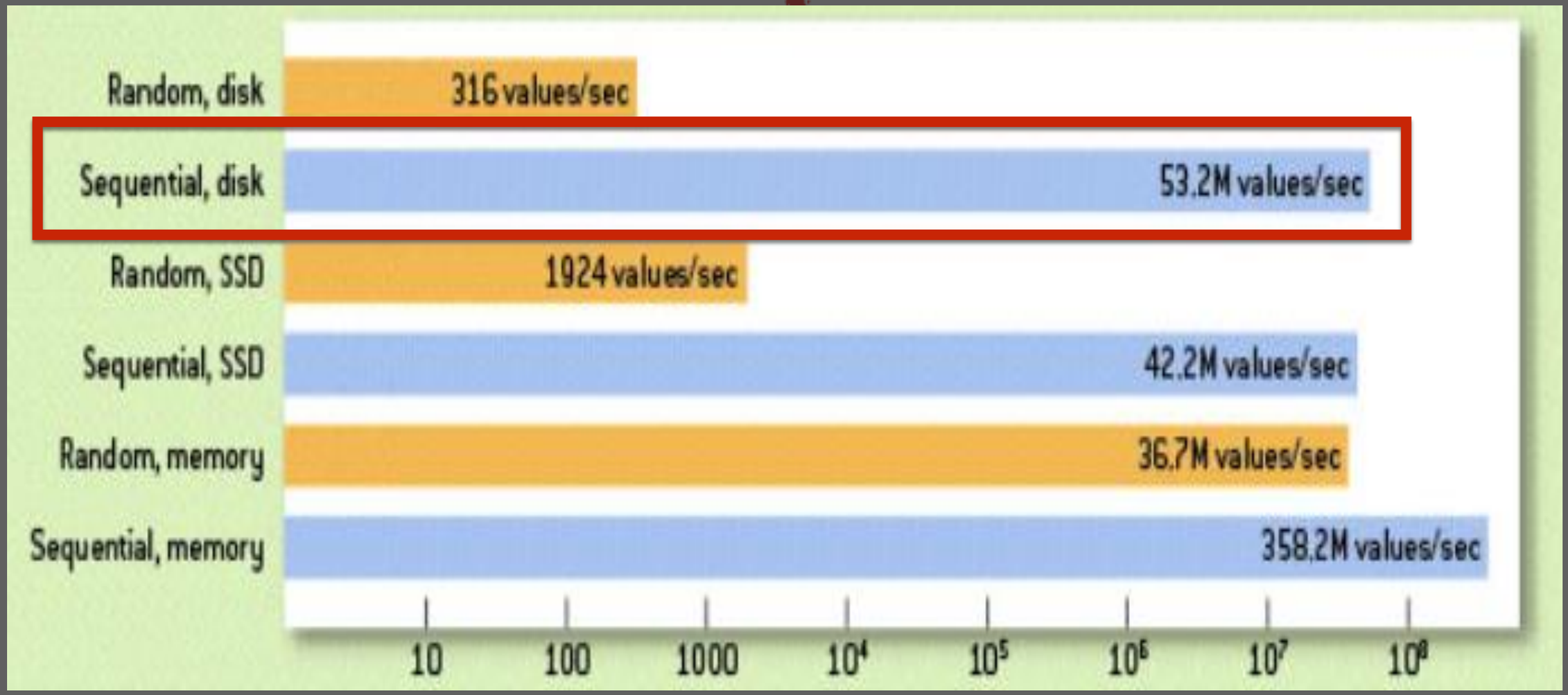
RabbitMQ

- 

Message Queue는 어떤것을 사용할까?

APACHE kafka™
A distributed streaming platform

RabbitMQ

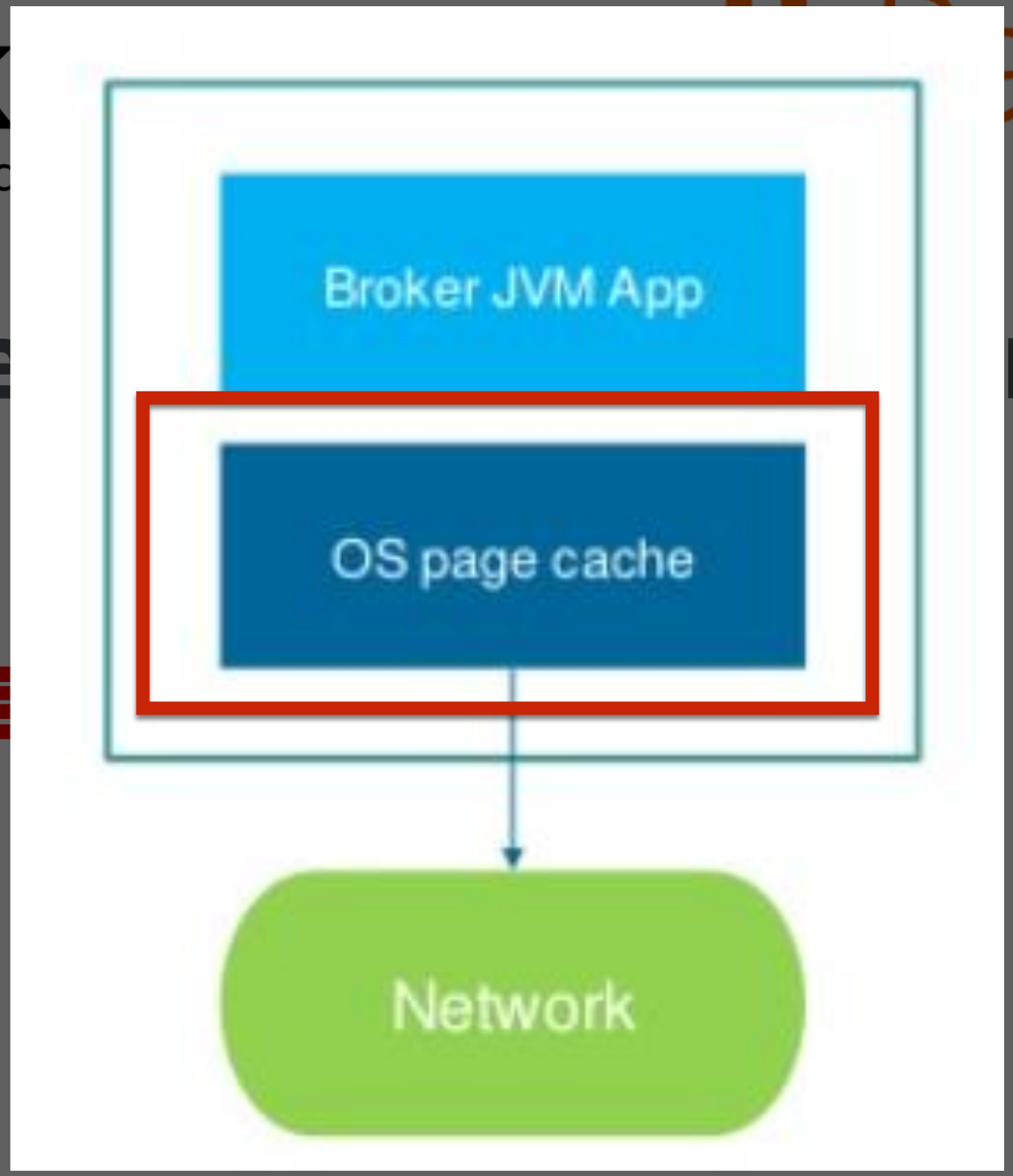


Message Queue는 어떤것을 사용할까?

APACHE kafka
A distributed

RabbitMQ

- FileSyste ry에 저장
- TCP/IP
- 뛰어난 성능 기능

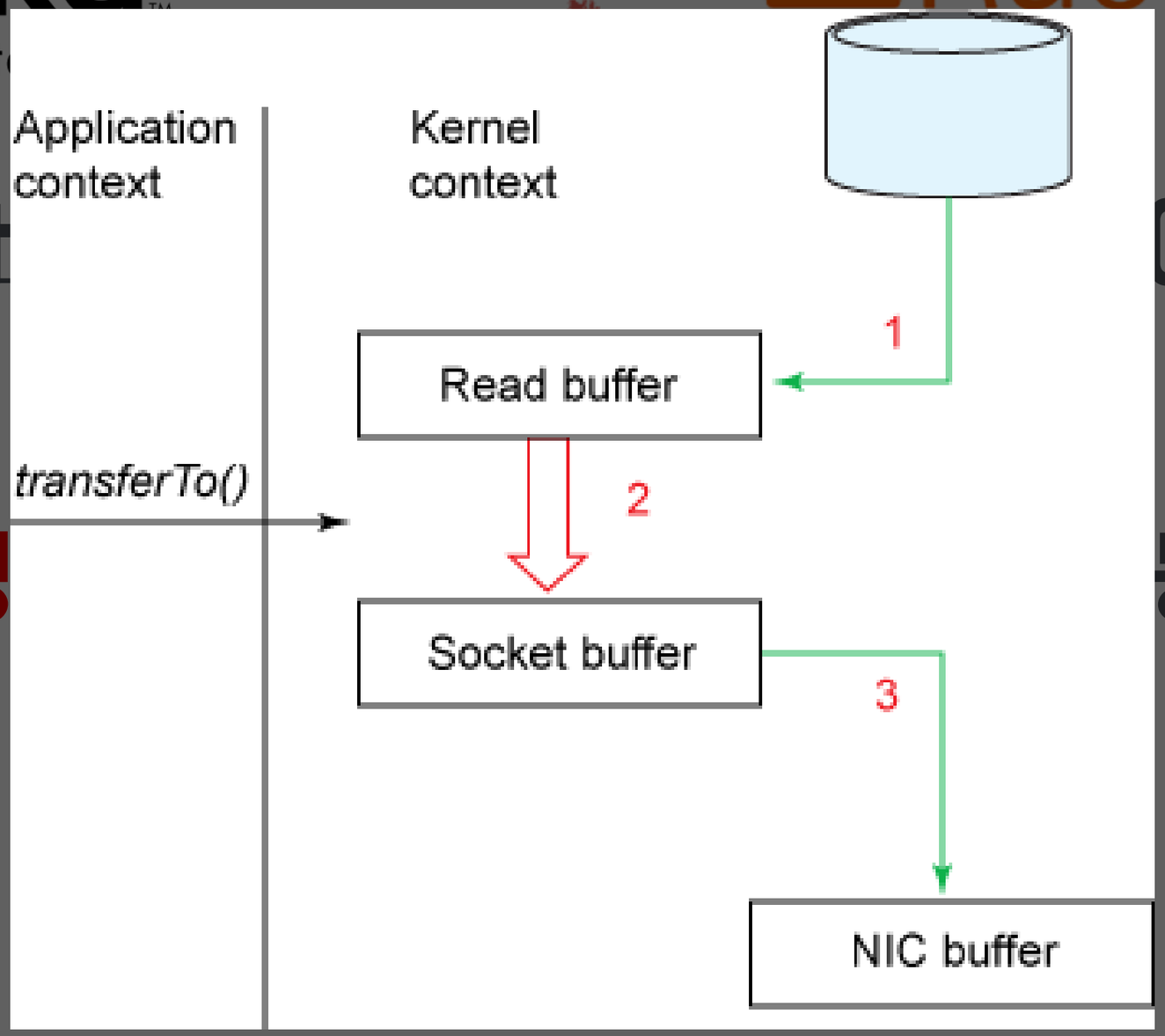


Message Queue는 어떤것을 사용할까?

APACHE kafka
A distribut

RabbitMQ

- FileSyst 에 저장
- TCP/IP
- 뛰어난 성 이



Message Queue는 어떤것을 사용할까?

- FileSystem에 저장
- TCP/IP
- 뛰어난 성능

- Memory에 저장
- AMQP
- 다양한 기능

Message Queue는 어떤것을 사용할까?

- FileSystem에 저장
- TCP/IP
- 뛰어난 성능
- 최소 1번은 메시지 전달을 보장함

- Memory에 저장
- AMQP
- 다양한 기능
- 자체적으로 Clustering을 지원

Message Queue는 어떤것을 사용할까?

- FileSystem에 저장
- TCP/IP
- 뛰어난 성능
- 최소 1번은 메시지 전달을 보장함

- Memory에 저장
- AMQP
- 다양한 기능
- 자체적으로 Clustering을 지원

Spring Cloud Stream사용법

Spring Cloud Stream사용법

- http://start.spring.io

SPRING INITIALIZR bootstrap your application now

Generate a Maven Project with Spring Boot 1.5.2

Project Metadata

Artifact coordinates

Group

com.example

Artifact

demo

Dependencies

Add Spring Boot Starters and dependencies to your application

Search for dependencies

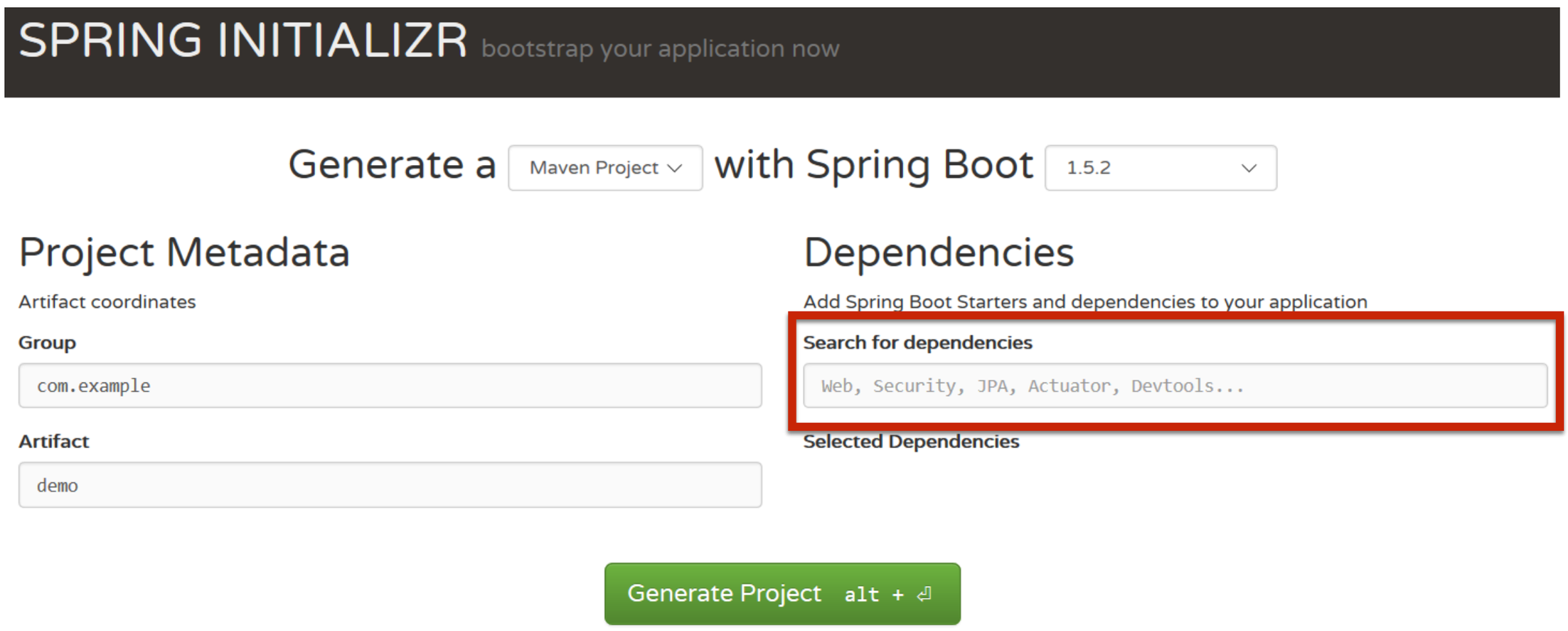
Web, Security, JPA, Actuator, Devtools...

Selected Dependencies

Generate Project alt + ⏎

Spring Cloud Stream사용법

- Stream kafka 라고 입력



Spring Cloud Stream사용법

- Stream kafka 추가 완료

SPRING INITIALIZR bootstrap your application now

Generate a Maven Project with Spring Boot 1.5.2

Project Metadata

Artifact coordinates

Group

com.example

Artifact

demo

Dependencies

Add Spring Boot Starters and dependencies to your application

Search for dependencies

Web, Security, JPA, Actuator, Devtools...

Selected Dependencies

Stream Kafka

Generate Project alt + ⏎

Spring Cloud Stream사용법

- Generate Project를 이용해서 생성

SPRING INITIALIZR bootstrap your application now

Generate a Maven Project with Spring Boot 1.5.2

Project Metadata

Artifact coordinates

Group

com.example

Artifact

demo

Dependencies

Add Spring Boot Starters and dependencies to your application

Search for dependencies

Web, Security, JPA, Actuator, Devtools...

Selected Dependencies

Stream Kafka

Generate Project alt + ⏎

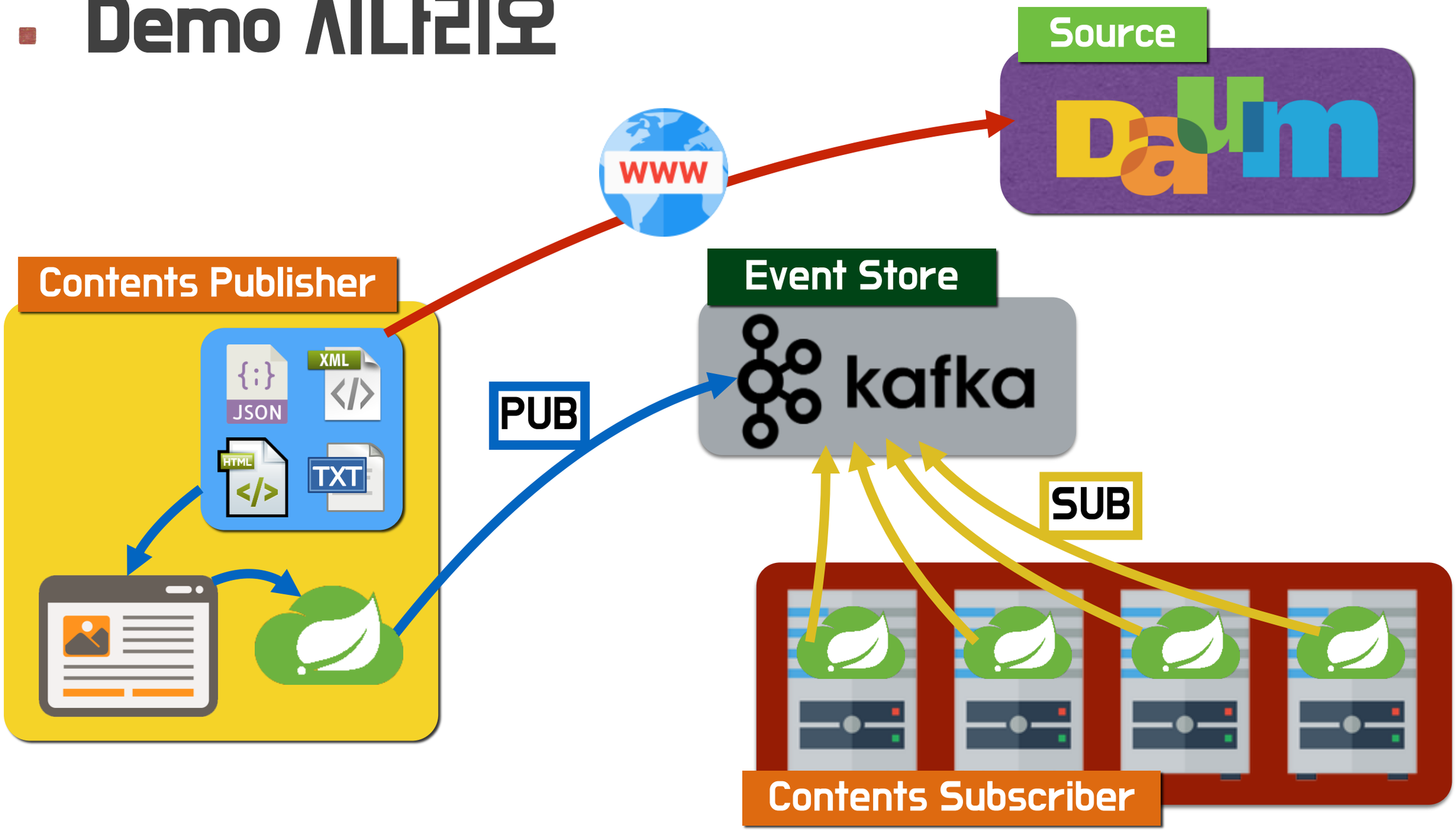
kakao

Spring Cloud Stream Demo

- Demo

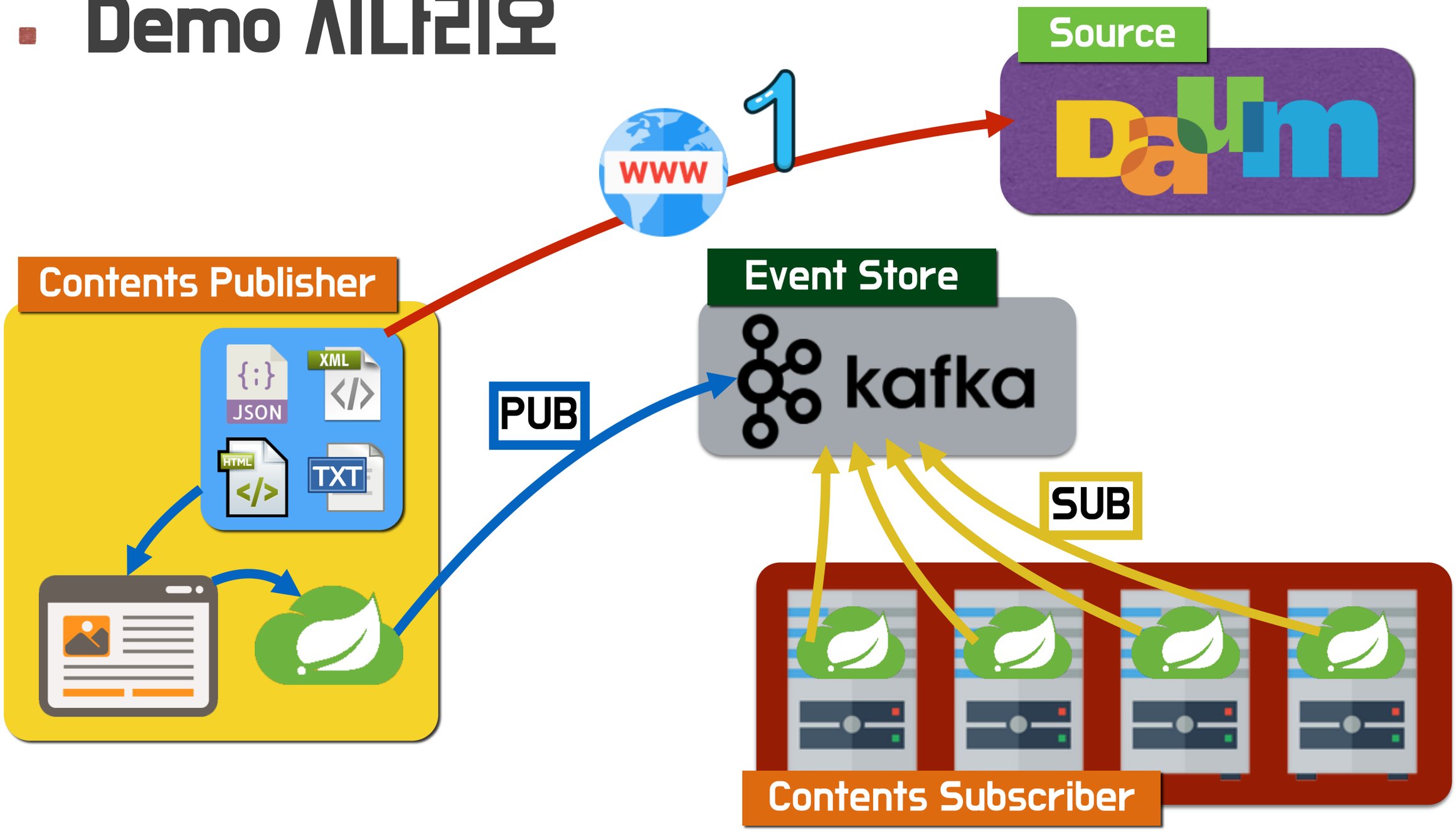
Spring Cloud Stream Demo

- Demo 시나리오



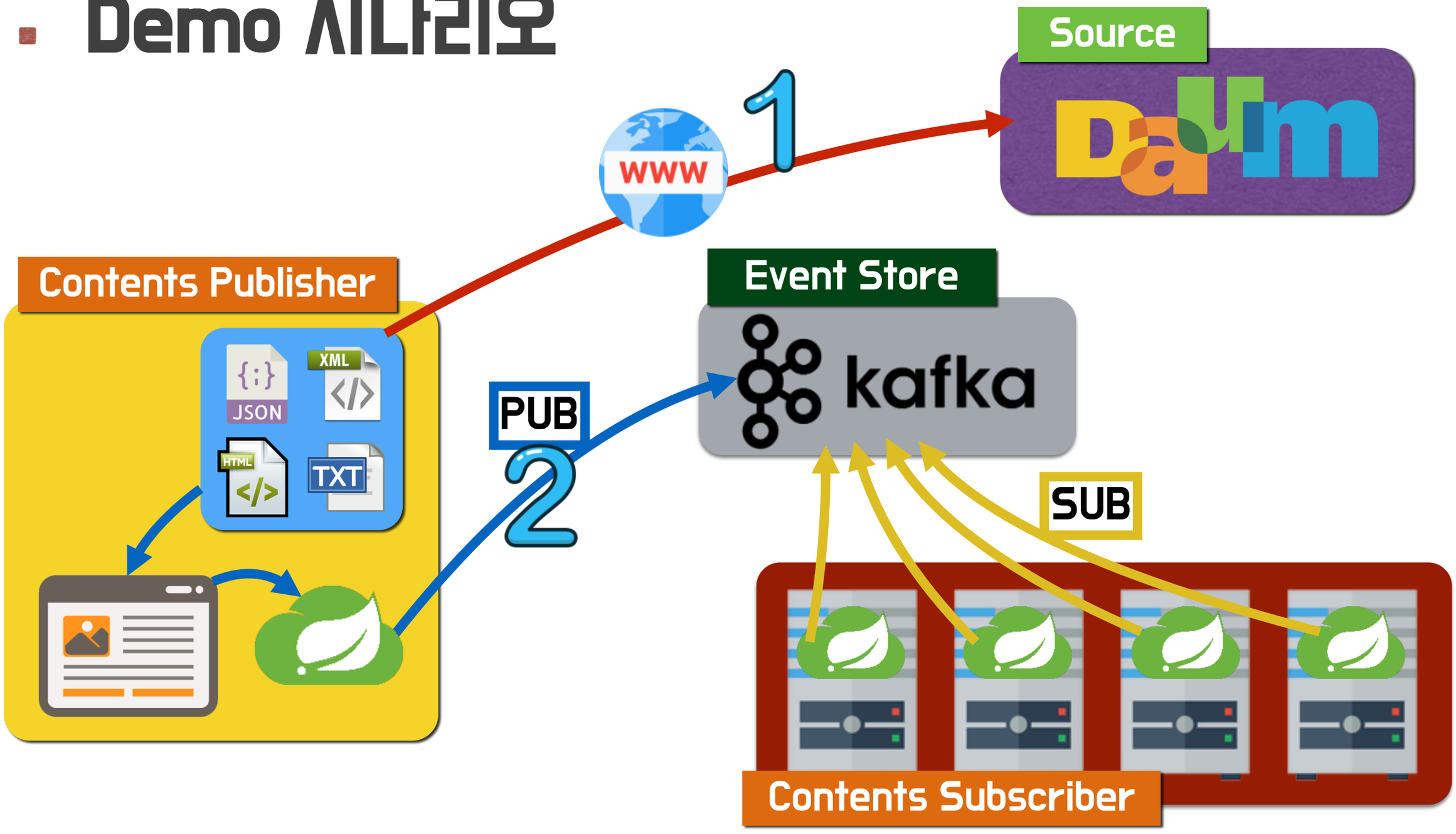
Spring Cloud Stream Demo

- Demo 시나리오



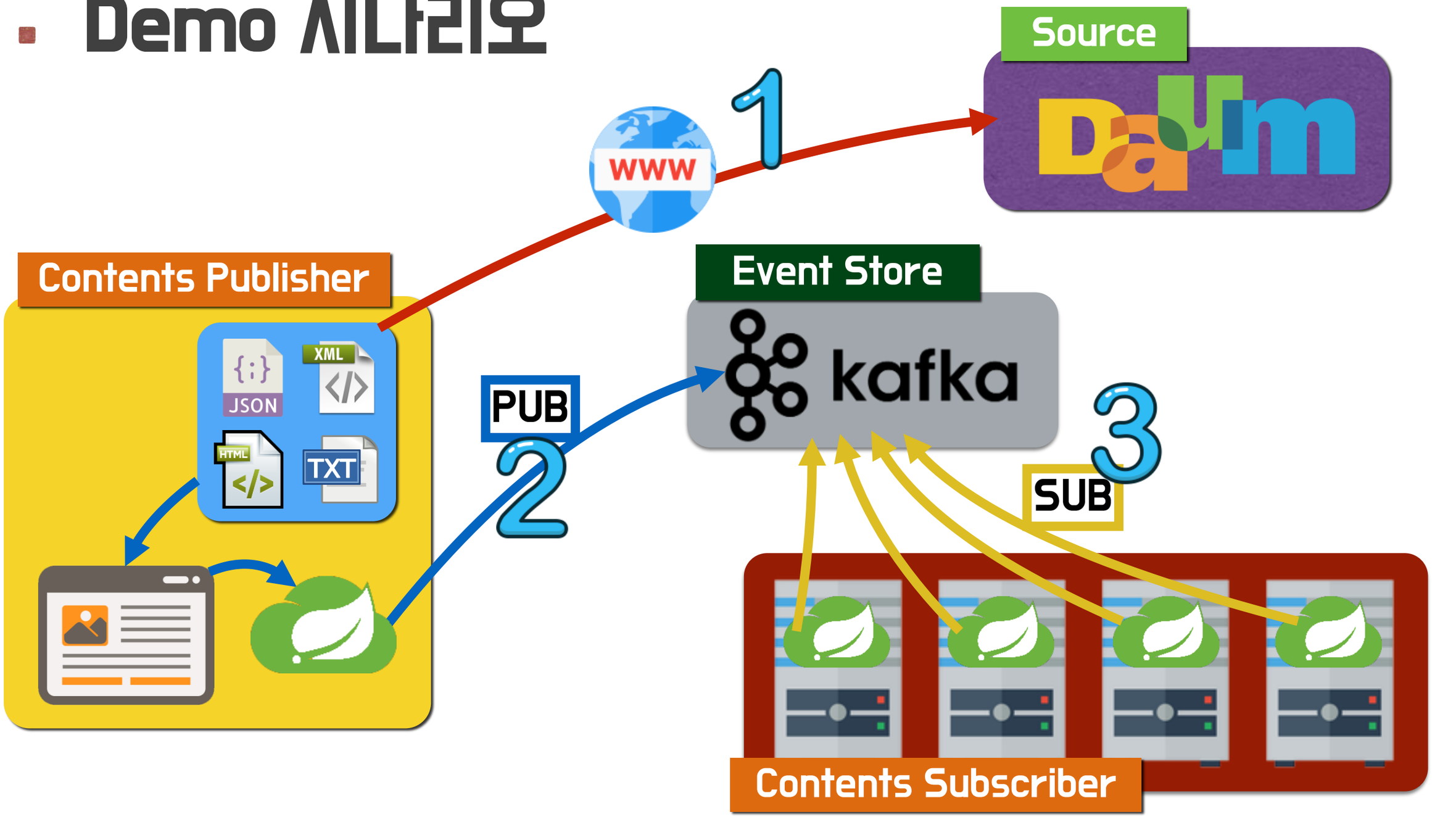
Spring Cloud Stream Demo

- Demo 시나리오



Spring Cloud Stream Demo

- Demo 시나리오



정리를 하면

- 배포방식 Push방식 -> Pub/Sub방식 개선

정리를 하면

- 배포방식 Push방식 -> Pub/Sub방식 개선

  -> 대폭적인 성능향상, 유연한 확장 가능

정리를 하면

- 배포방식 Push방식 -> Pub/Sub방식 개선

  -> 대폭적인 성능향상, 유연한 확장 가능

  -> Infra관점의 관리 포인트 증가

정리를 하면

- Spring Cloud Stream사용

정리를 하면

- Spring Cloud Stream사용

  -> 간편한 에너테이션 덕분에 빠른 개발이 가능

정리를 하면

- Spring Cloud Stream사용

  -> 간편한 에너테이션 덕분에 빠른 개발이 가능

  -> 개발자는 비즈니스 로직에 집중할 수 있음

정리를 하면

- Spring Cloud Stream사용
  - -> 간편한 에너테이션 덕분에 빠른 개발이 가능
  - -> 개발자는 비즈니스 로직에 집중할 수 있음
  - -> 뭔가 단점을 말하고 싶은데 잘 못 찾겠음...

(Spring에서만 사용 가능하다 정도?)

Kafka 1원 팁

- 만약 Kafka를 직접 운영한다면?

  -> Kafka Manager (Kafka UI 관리 툴)

  (https://github.com/yahoo/kafka-manager)

Kafka 1원 팁

- 만약 Kafka를 직접 운영한다면?

  -> Kafka Manager (Kafka UI 관리 툴)

  (https://github.com/yahoo/kafka-manager)

  -> Kafka의 Heap Memory는 크게 잡지 말 것

  (32G Memory 서버 기준 6G정도)

Kafka 1원 팁

- 만약 Kafka를 직접 운영한다면?

  -> Kafka Manager (Kafka UI 관리 툴)

  (https://github.com/yahoo/kafka-manager)

  -> Kafka의 Heap Memory는 크게 잡지 말 것

  (32G Memory 서버 기준 6G정도)

  -> num.network.threads설정을 확인할 것

  (Request가 늘어나면 적당한 값으로 조정필요)

RabbitMQ는 어떨까요?

- 만약 RabbitMQ를 직접 운영한다면?

RabbitMQ는 어떨까요?

- 만약 RabbitMQ를 직접 운영한다면?

  -> 잘 모르겠음.. Google에 검색해보세요..

Spring Cloud Stream 1업 팁

Spring Cloud Stream 1업 팁

- 운영 중 신규 Consumer를 추가 투입 시

Spring Cloud Stream 1원 팁

- 운영 중 신규 Consumer를 추가 투입 시

  -> 기존 메시지를 한꺼번에 Subscribe함

Spring Cloud Stream 1원 팁

- 운영 중 신규 Consumer를 추가 투입 시

  -> 기존 메시지를 한꺼번에 Subscribe함

  (Index는 Consumer가 가지고 있음)

Spring Cloud Stream 1원 팁

- 운영 중 신규 Consumer를 추가 투입 시

  -> 기존 메시지를 한꺼번에 Subscribe함

  (Index는 Consumer가 가지고 있음)

  -> 새로 유입되는 메시지부터 받기를 원하면

Spring Cloud Stream 1원 팁

- 운영 중 신규 Consumer를 추가 투입 시

  -> 기존 메시지를 한꺼번에 Subscribe함

  (Index는 Consumer가 가지고 있음)

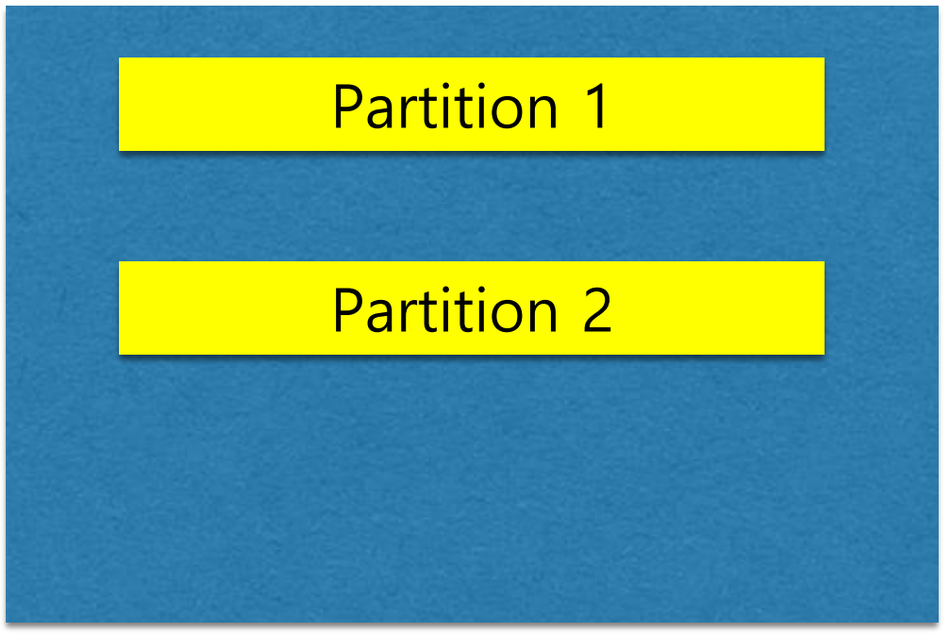
  -> 새로 유입되는 메시지부터 받기를 원하면

  -> autoOffsetReset : latest

Spring Cloud Stream 1업 팁

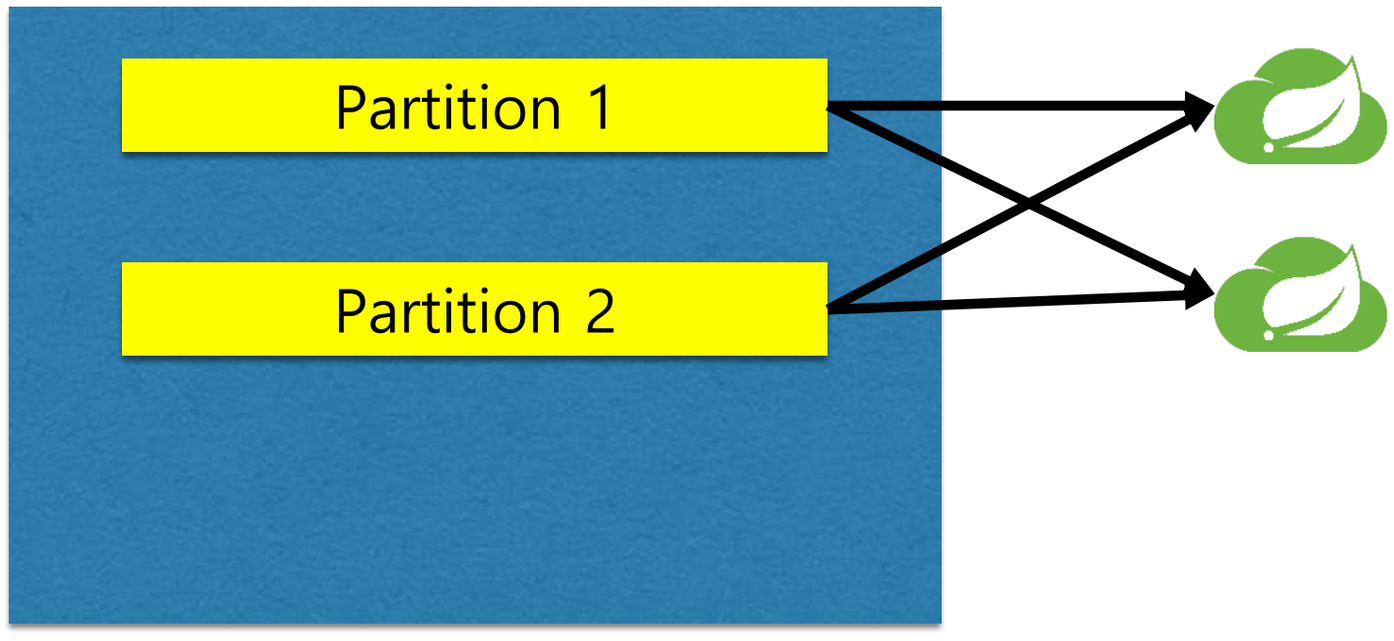
Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우



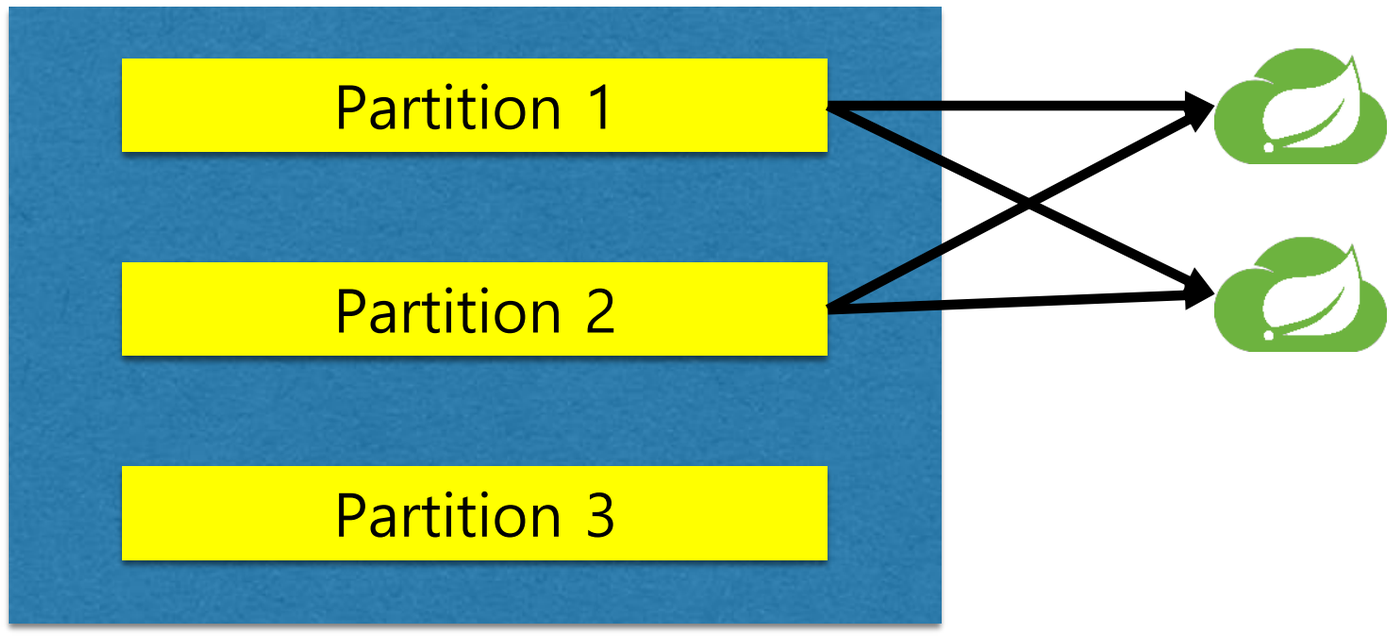
Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우
- 기본설정은 모든 파티션에 Assign



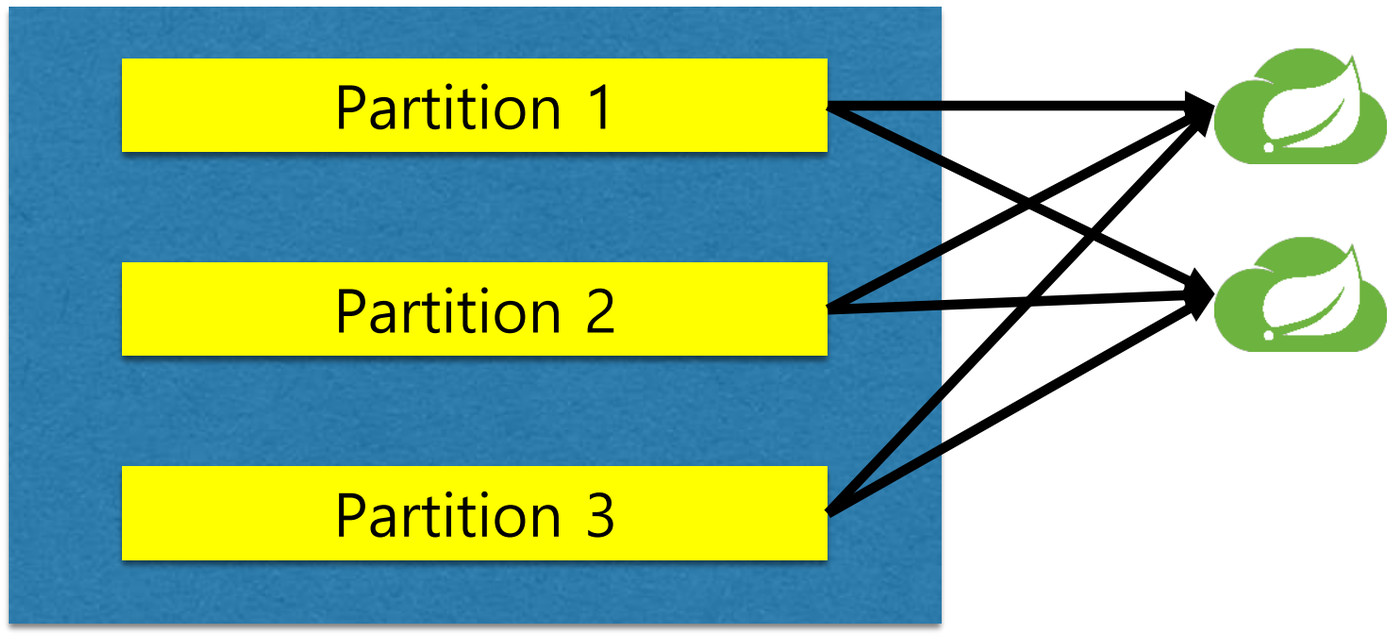
Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우
- 기본설정은 모든 파티션에 Assign



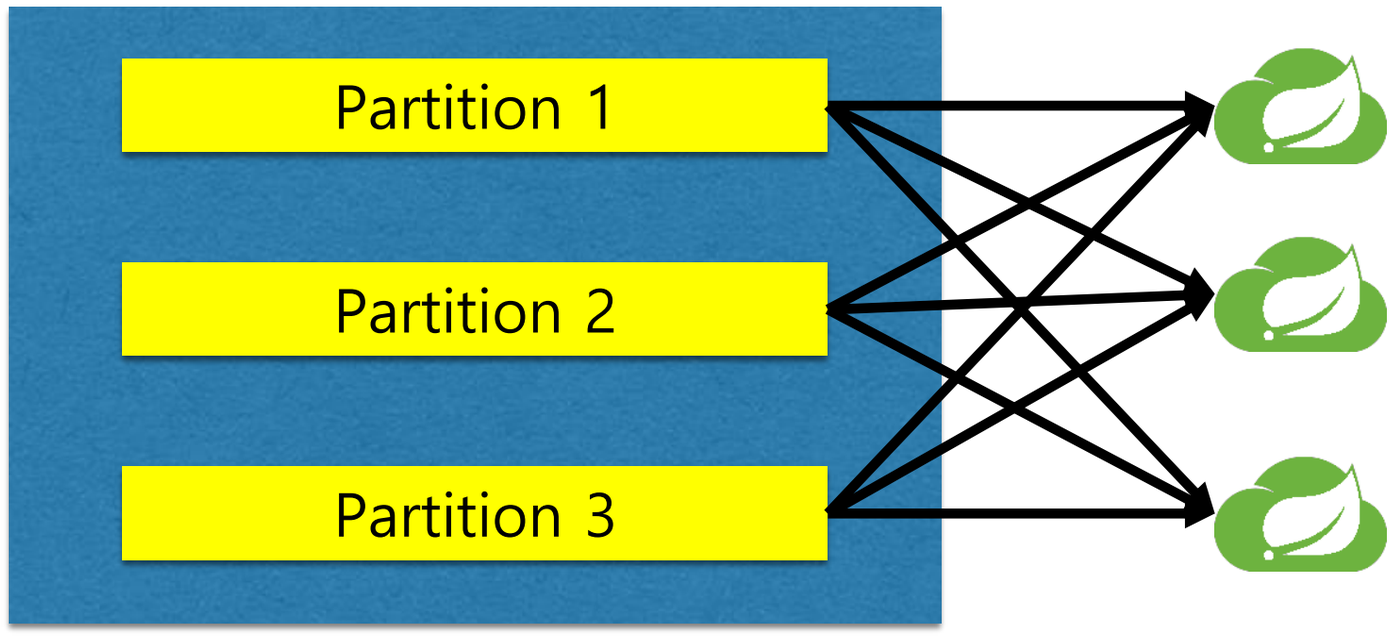
Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우
- 기본설정은 모든 파티션에 Assign



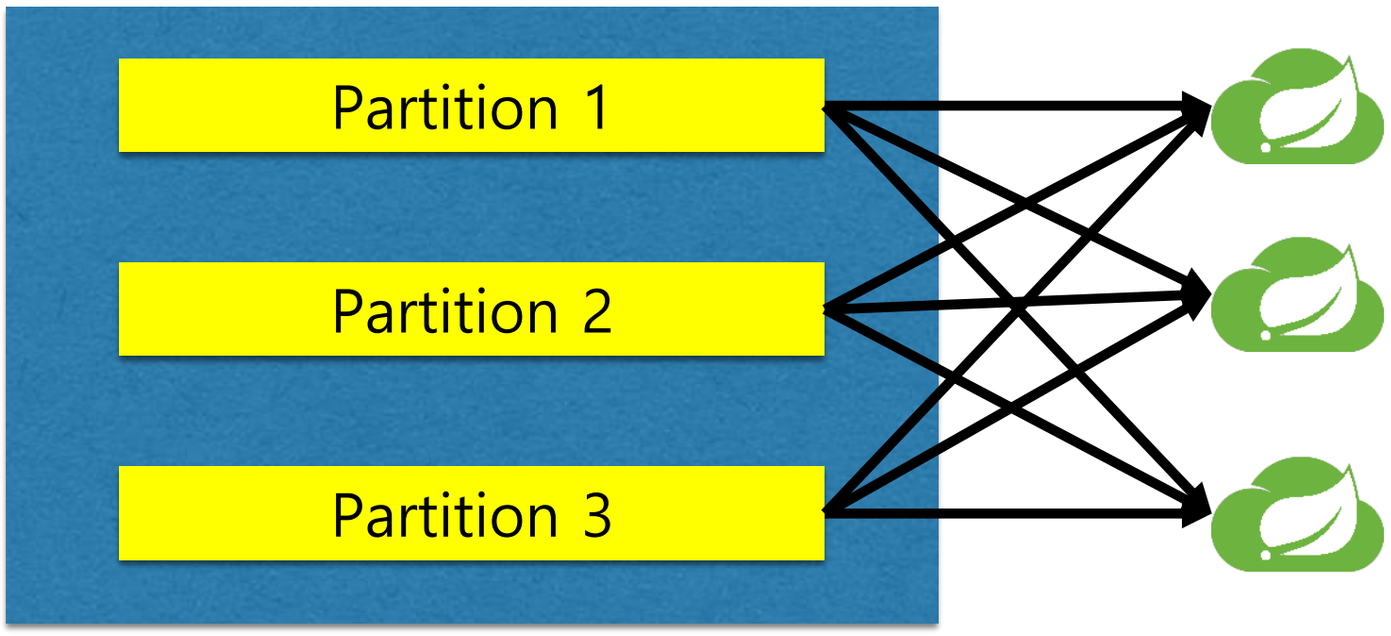
Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우
- 기본설정은 모든 파티션에 Assign



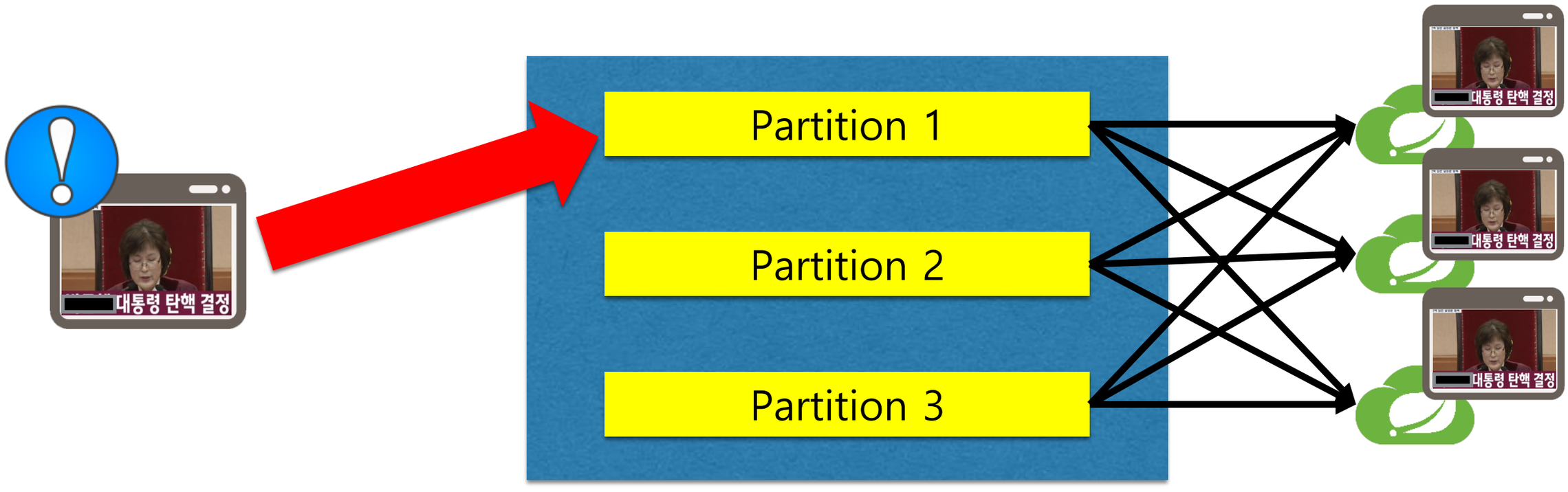
Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우
- 기본설정은 모든 파티션에 Assign



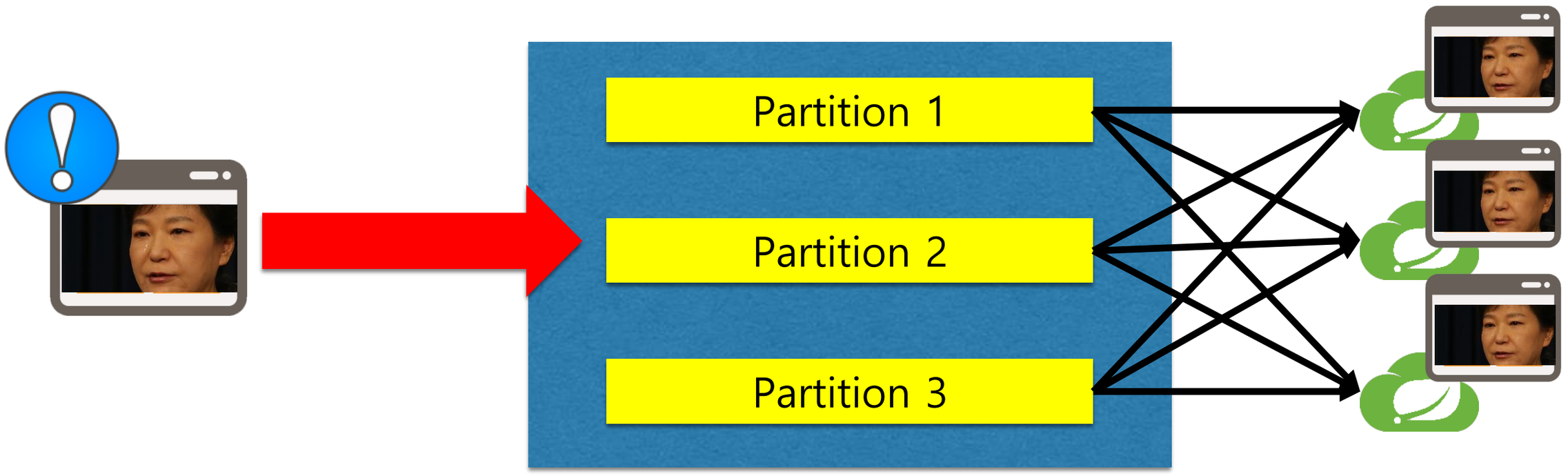
Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우
- 기본설정은 모든 파티션에 Assign
- 이때 콘텐츠를 Publish하면?



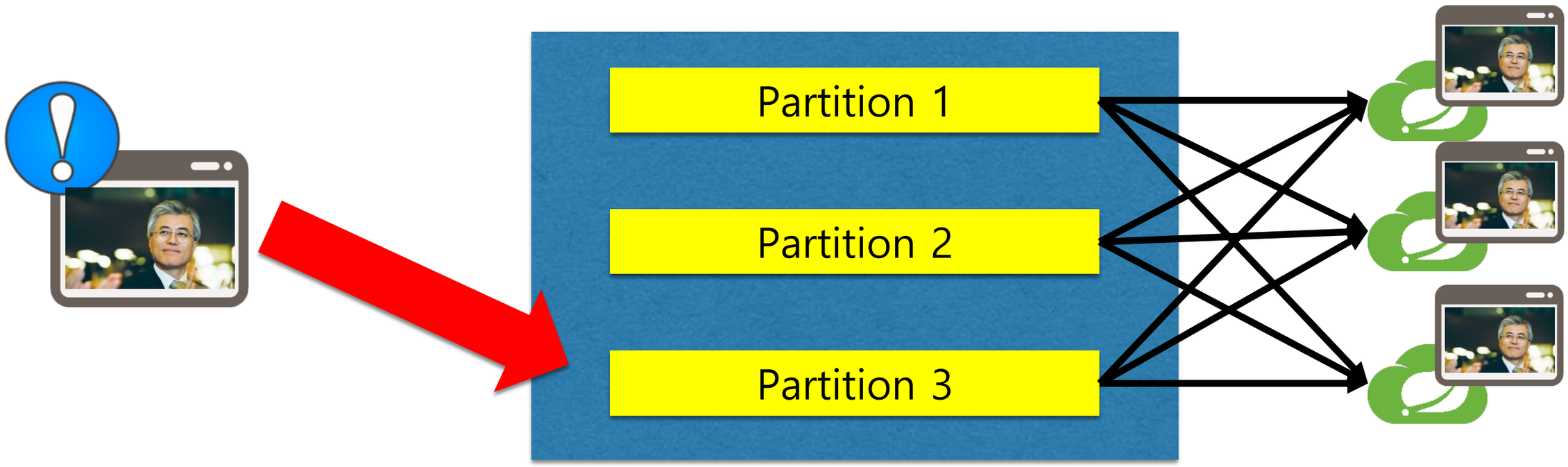
Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우
- 기본설정은 모든 파티션에 Assign
- 이때 콘텐츠를 Publish하면?



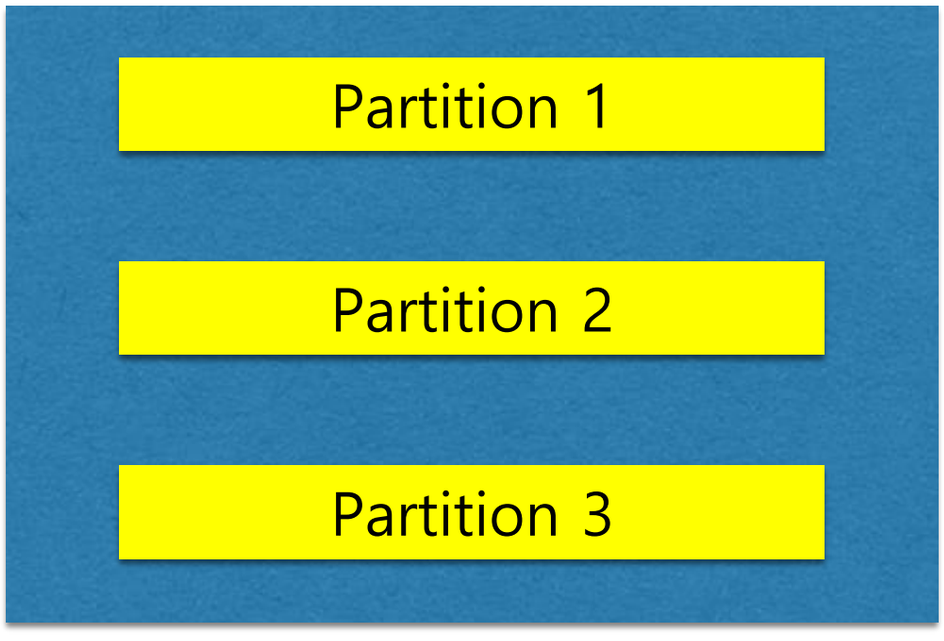
Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우
- 기본설정은 모든 파티션에 Assign
- 이때 콘텐츠를 Publish하면?



Spring Cloud Stream 1원 팁

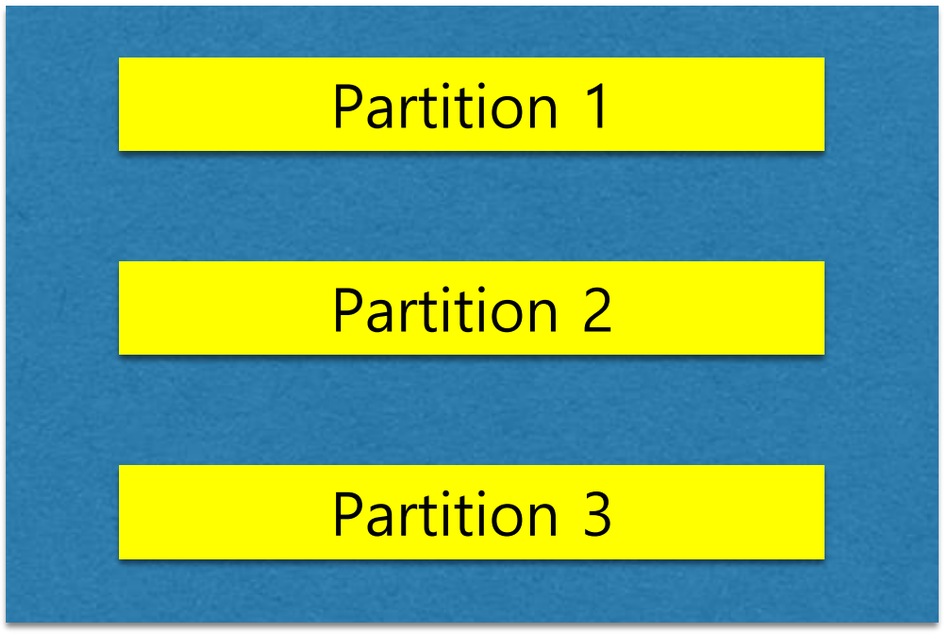
- Topic의 다중 Partition을 사용할 경우
- 기본설정은 모든 파티션에 Assign
- 처리량을 증가시키려면?



Spring Cloud Stream 1원 팁

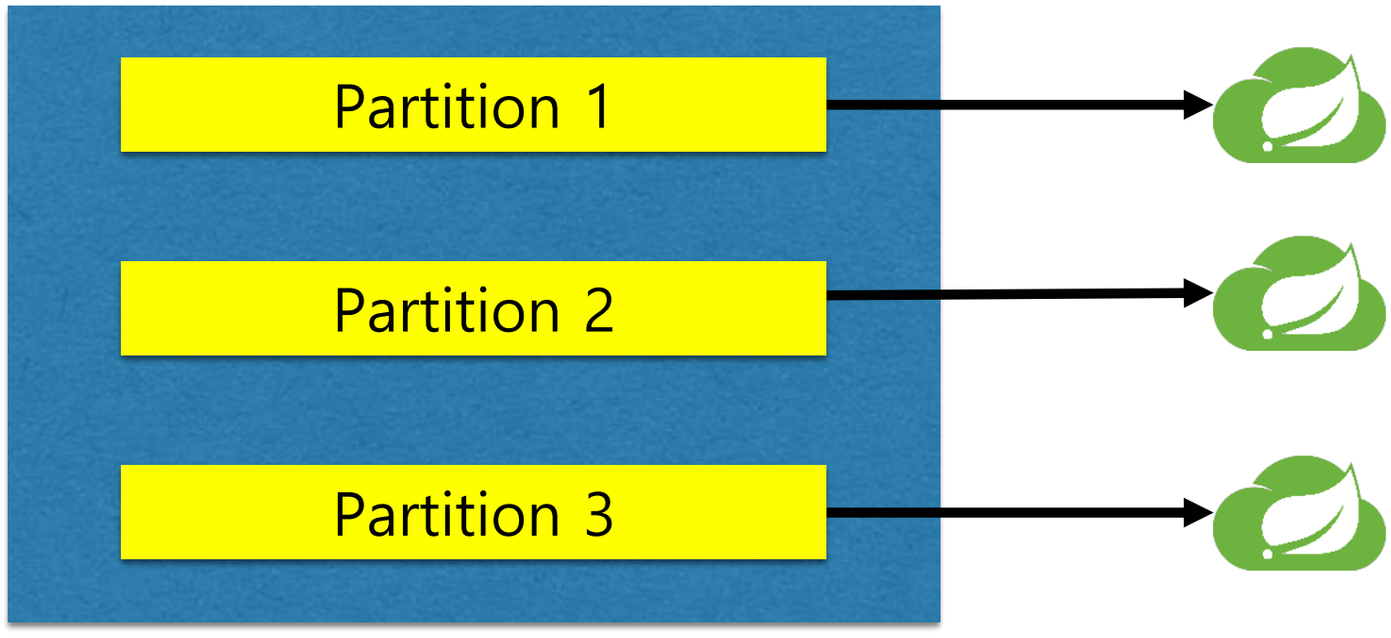
- Topic의 다중 Partition을 사용할 경우

  -> ConsumerGroup 사용



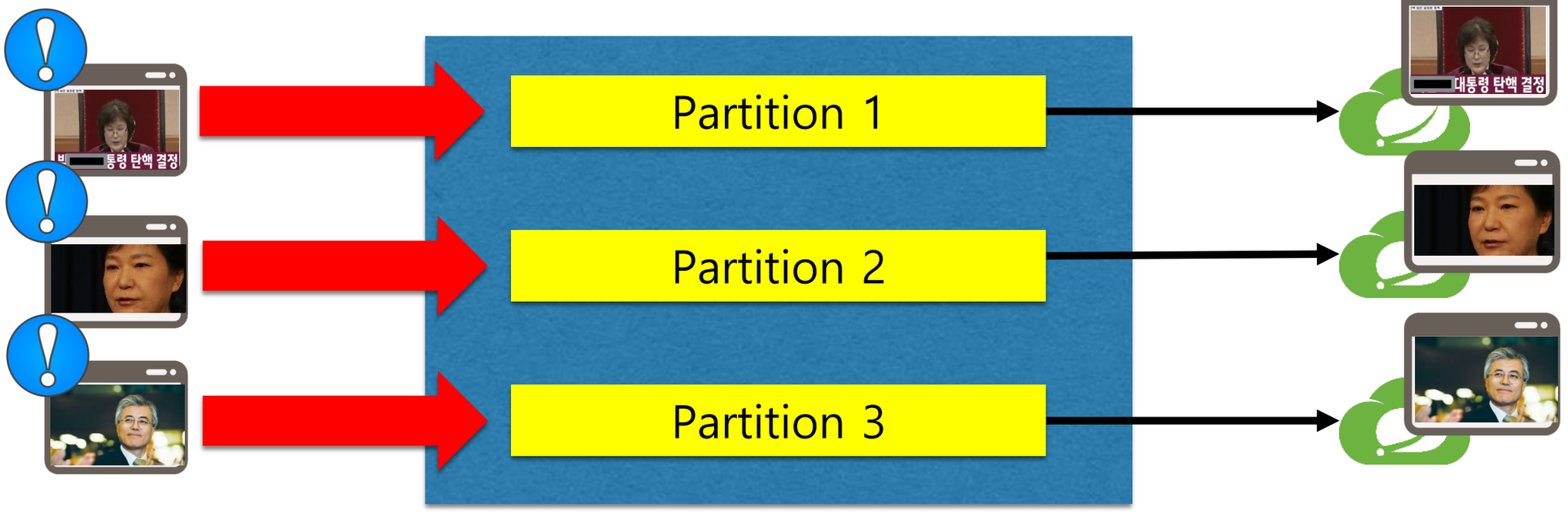
Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우
  - -> ConsumerGroup 사용
  - -> Consumer별로 다른 파티션에 Assign가능



Spring Cloud Stream 1원 팁

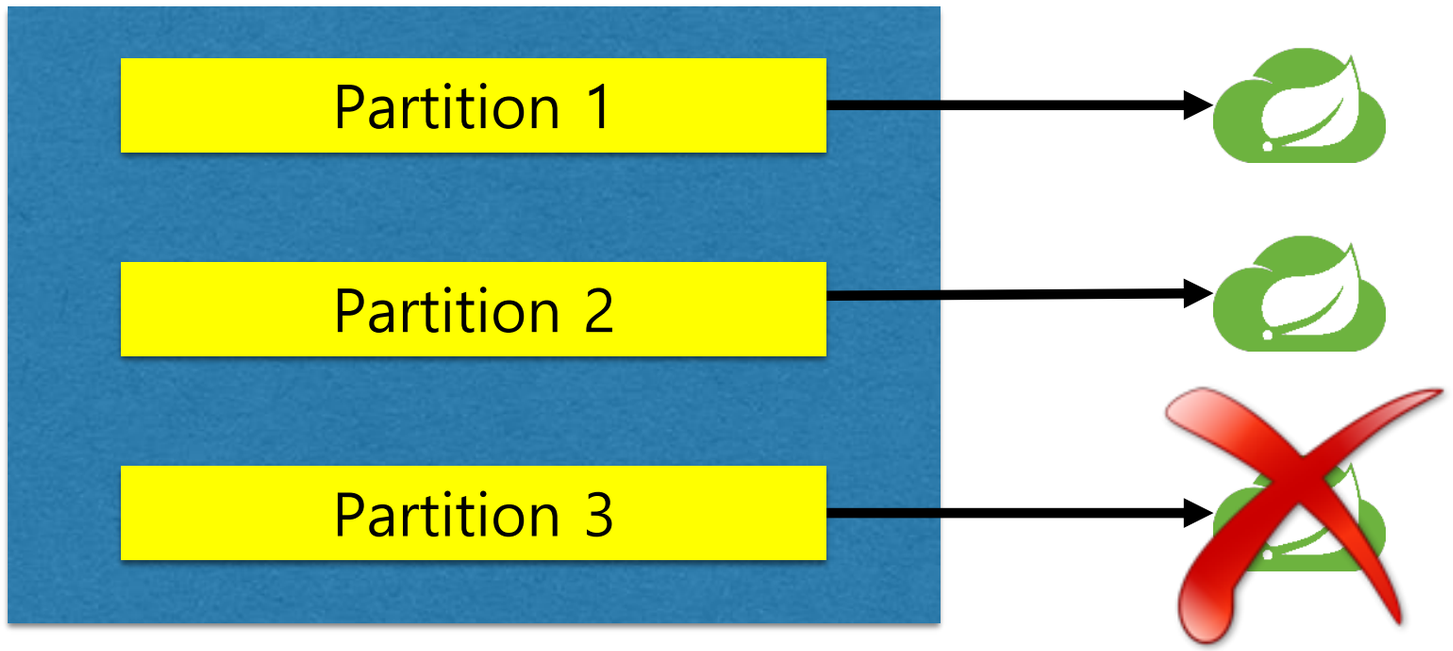
- Topic의 다중 Partition을 사용할 경우
  - -> ConsumerGroup 사용
  - -> Consumer별로 다른 파티션에 Assign가능



Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우

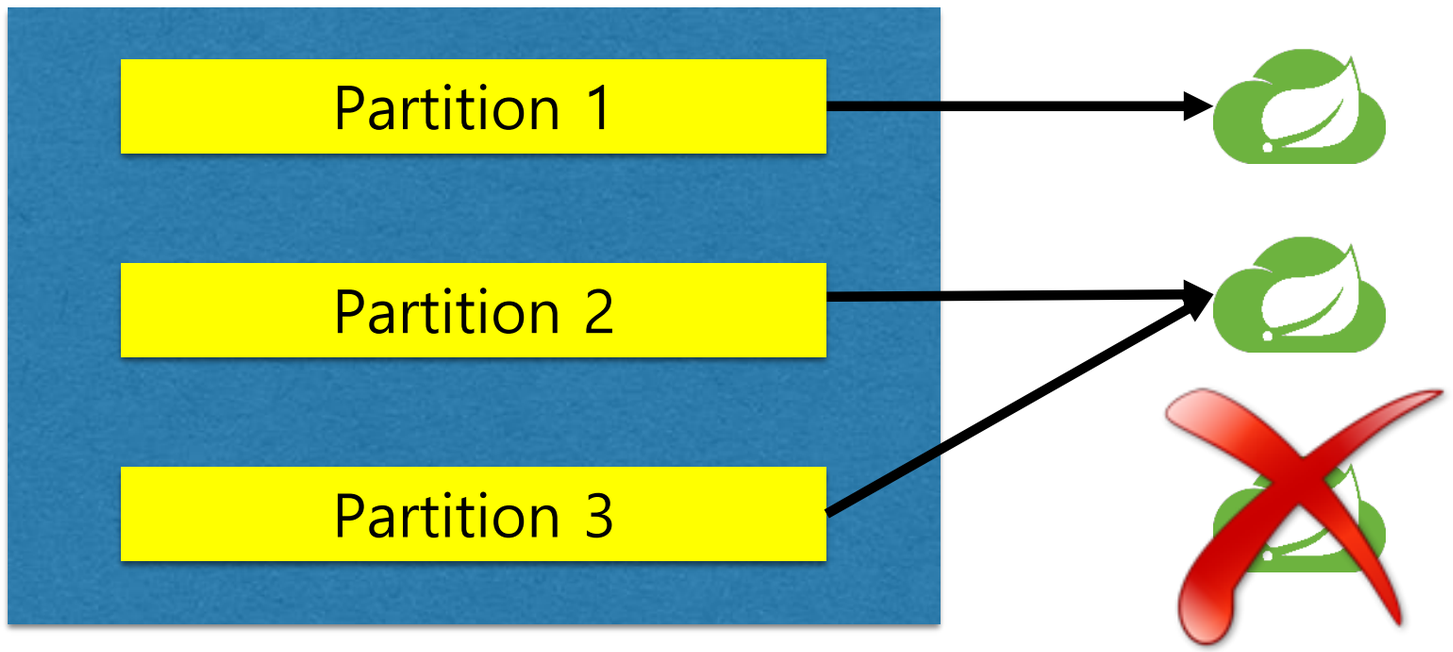
  -> 만약 Consumer가 죽으면?



Spring Cloud Stream 1원 팁

- Topic의 다중 Partition을 사용할 경우

  -> 배포없이 살아있는 Consumer에 Assign됨



Spring Cloud Stream 1원 팁

- 다양한 옵션 활용이 가능합니다.

Spring Cloud Stream 1업 팁

- 다양한 옵션 활용이 가능합니다.
- 사용방법은..

Spring Cloud Stream 1업 팁

- 다양한 옵션 활용이 가능합니다.
- 사용방법은.. 시간이 없어 생략..

참고사항

- 같이 쓰면 좋은 Spring Cloud Component
  - -> Eureka : Service Discovery & Registry
  - -> Config Server : Dynamic한 설정의 분리

참고사항

- 같이 쓰면 좋은 Spring Cloud Component
  - -> Eureka : Service Discovery & Registry
  - -> Config Server : Dynamic한 설정의 분리

키워드 Remind

- Spring Cloud Stream
- Spring Cloud & Spring Boot
- Stream & Batch
- Pub/Sub - Publish & Subscribe
- Message Queue (Kafka, RabbitMQ)

마치며...

- 콘텐츠 배포 시스템은 Spring Cloud Stream 의 한가지 활용 예일 뿐입니다.

마치며...

- 콘텐츠 배포 시스템은 Spring Cloud Stream 의 한가지 활용 예일 뿐입니다.
- Pub/Sub이 필요한 다양한 Service에서 활용 가능합니다.

마치며...

- 콘텐츠 배포 시스템은 Spring Cloud Stream 의 한가지 활용 예일 뿐입니다.
- Pub/Sub이 필요한 다양한 Service에서 활용 가능합니다.
- 다양한 레퍼런스를 기대합니다.

Thank you.

kakao

- # Thank you.
- 카카오 쇼핑플랫폼개발셀에서 커머스플랫폼 개발자를 모집하고 있습니다.

  -> 저와 같이 노가다 하실분? (leekyoungil@gmail.com)