Apache Kafka

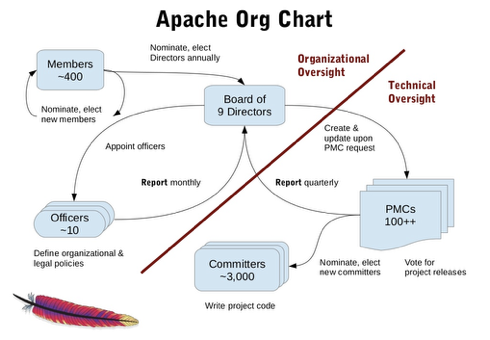
미들웨어운영팀 박재연 사원

목차

1. Apache Foundation
2. 개요
3. 재단 운영 방식
4. 대표적인 프로젝트
5. Apache Kafka
6. 개요
7. 주요 인원 소개
8. 주요 기능 및 코어 메커니즘
9. 활용 방안
10. 미들웨어에서 Kafka의 역할
11. Apache Foundation
12. 개요

|  |  |
| --- | --- |
| 설립 목적 |  |
| 설립 시기 | 1999년 6월 |
| 설립 이념 |  |
| 대표 프로젝트 | Apache HTTP Server, Apache Hadoop, Apache Tomcat, Apache Lucene 등 |
| 웹사이트 | [www.apache.org](http://www.apache.org) |

1. 재단 운영 방식



Apache License

아파치 재단이나, 재단 내 프로젝트로 만들어진 모든 소프트웨어는 아파치 라이선스 2.0에 의해 배포.

2.0 버전으로 개정하면서, 라이선스 수정 없이 재사용할 수 있고, 기여자들의 특허 라이선스

Apache Way

아파치 재단의 소프트웨어는 리눅스, 파이썬 등처럼 특정 인물을 중심으로 개발되는 것이 아니라, 같은 관심을 가진 집단으로 개발된다.

|  |  |
| --- | --- |
| 분류 | 설명 |
| User |  |
| Contributor |  |
| Committer |  |
| PMC |  |

1. 대표적인 프로젝트

차트로 가장 많이 개발되고 있는 카테고리

탑레벨과 인큐베이팅 차이

Total number of projects + sub-projects - 328 (not including Apache Labs initiatives)

Top-Level Projects - 198

Podlings in the Apache Incubator - 51

Other groups, including operations/support - 62

Top 5 Apache Project Repositories by Commits

1. Hadoop
2. HBase
3. Beam
4. Camel
5. Flink
6. Apache Kafka
7. 유래 및 목표

* 2011년, LinkedIn에서 개발된 **비동기 방식 분산 메시징 시스템**으로,   
  2014년에 아파치를 통해 오픈 소스화. 현재는 컨플루언트라는 회사.
* LinkedIn 사내에서 포털의 사용자 활동 데이터를 수집하고, 사용자에게 적절한 정보를 제공.
* 스칼라를 사용하여 제작
* 카프카의 제작 목표
  + 메시지 프로듀서와 컨슈머 사이의 느슨한 연계
  + 다양한 형태의 데이터 사용 시나리오와 장애 처리 지원을 위한 메시지 데이터 유지
  + 빠른 처리 시간을 지원하는 구성 요소로 시스템의 전반적 처리량 최대화
  + 이진 데이터 형식을 사용하여 다양한 데이터 형식과 유형 관리
  + 기존 클러스터 구성에 영향을 주지 않고 일정한 서버의 확장성 지원
* 분산 시스템 지원하는 Pub/Sub 메시징 시스템으로 구현.

\* P2P 메시징 모델과 Pub/Sub 메시징 모델의 차이

|  |  |
| --- | --- |
|  |  |
| P2P 메시징 모델 | Pub/Sub(발행/구독) 메시징 모델 |
| 한지점에서 다른 지점으로 메시지를 전송한다. 하나 이상의 컨슈머가 메시지를 읽을 수 있지만, 특정 메시지는 오직 하나의 컨슈머만 읽을 수 있다. 컨슈머가 읽은 메시지는 대기열에서 사라지기 때문이다. 이 모델을 사용 할 수 있는 대표적인 시스템이 주문 처리 시스템이다. | 하나의 메시지를 여러 컨슈머가 구독할 수 있다. 전형적인 뉴스구독 시스템에 어울리는 방식이다. 메시지를 읽으면서 큐에서 제거해 버리면 다른 컨슈머가 읽을 수 없으므로, 메시지는 큐에 계속 남아 있는다. 메시지가 늘어날 경우 저장공간이 부족하거나 데이터 처리의 효율이 떨어질 수 있으므로 일정 시간이 지난 메시지는 삭제한다. |

1. 주요 인원 소개

<https://kafka.apache.org/committers>

커미터로 등록된 사람들 목록 표시. 그 중 현재 컨플루언서에서 일하고 있는 주요 개발진.한 두줄씩

1. 카프카 특징

* 높은 쓰루풋과 낮은 레이턴시
* 기존 메시징 시스템 (ActiveMQ, RabbitMQ)등에서 브로커가 컨슈머에게 메시지를 Push하는 방식이었으나, 카프카에서는 컨슈머가 브로커에서 메시지를 Pull해가는 방식.
* 메모리에 직접 저장하는 것이 아닌 파일 시스템에 저장
* <https://epicdevs.com/17>

1. 카프카 구성 요소 및 구조

<https://devtimes.com/bigdata/2019/01/18/what-is-kafka/>

* + Topic, Partition, Offset

카프카에 저장되는 메시지는 Topic으로 구분되며, Topic은 여러개의 Partition으로 구분.

파티션 내 메시지 상대 위치를 나타내는 오프셋 정보를 통해 이전에 가져간 메시지 위치 정보를 알 수 있음

동시에 들어오는 데이터를 여러개 파티션에 나누어 저장하므로 병렬로 처리 가능.

* + Producer, Consumer

프로듀서는 메시지를 생산, 작성하는 주체, 컨슈머는 메시지를 소비, 읽는 주체.

* + Broker, ZooKeeper

브로커는 카프카 서버를 칭함

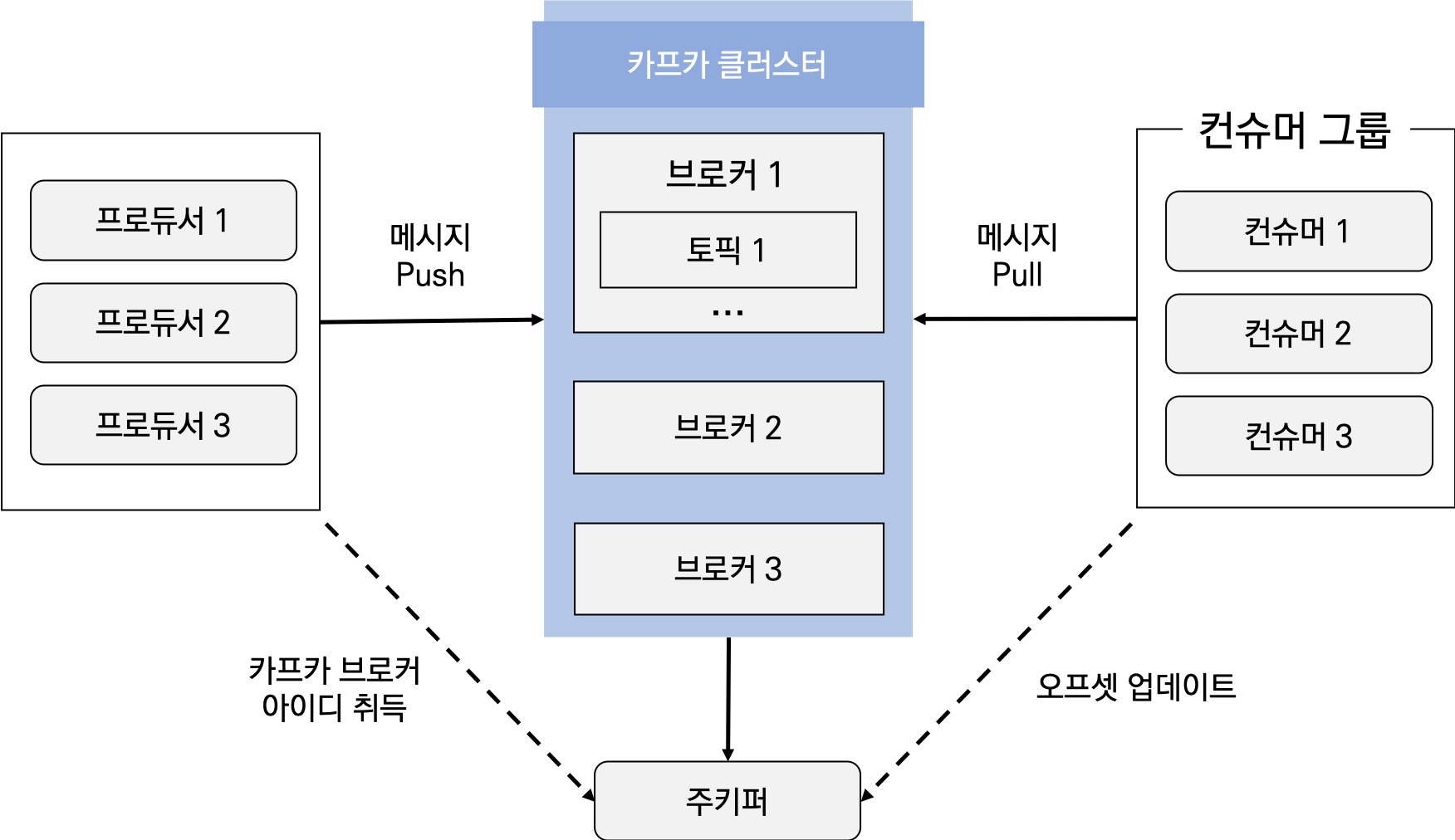
동일 노드 내에 여러개 브로커 서버를 올릴 수 있고, 주키퍼는 분산 메시지 큐 정보를 관리하는 역할.

카프카 실행을 위해서는 주키퍼 실행이 선행되어야 함

* + Replication

카프카에서는 레플리케이션 수를 임의로 지정하여 토픽을 만들 수 있음.

브로커에 문제가 생겼을 경우 해당 브로커의 역할을 다른 브로커에서 즉각적으로 대신 수행할 수 있게 하기 위함.



1. 활용 방안

* 메시징 시스템
* 웹사이트 활동 추적
* 로그 통합
* 스트리밍 프로세싱
* 로그 커밋

실제 활용 예로, 링크드인, 넷플릭스 설명

1. 미들웨어에서 Kafka의 역할

Mom으로서 미들웨어