Zookeeper

Zookeeper란 무엇인가?

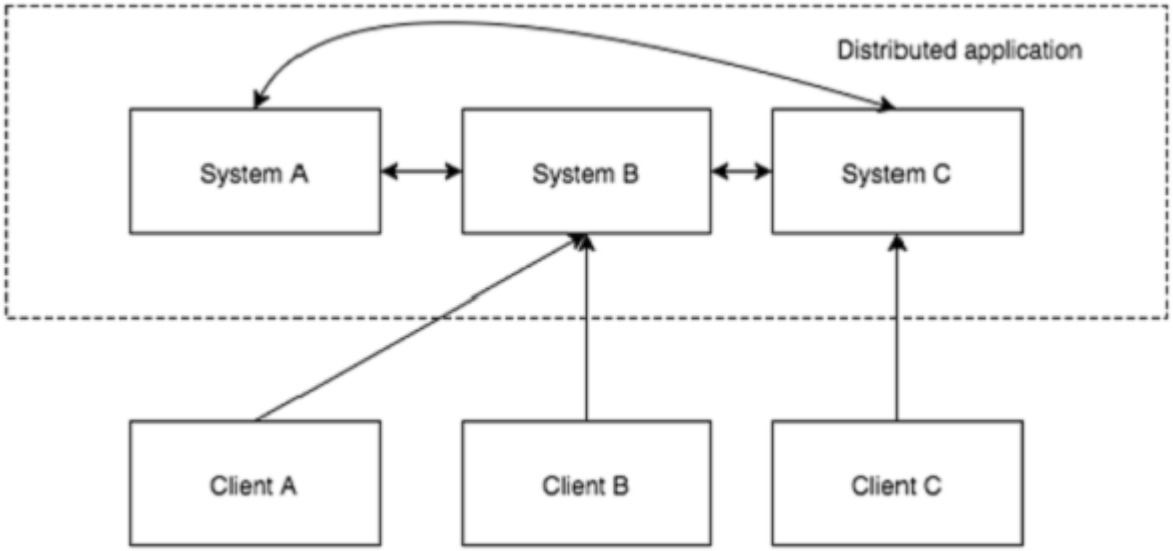
- 아파치 주키퍼(Apache Zookeepr)는 아파치 소프트웨어 재단 프로젝트중의 한 소프트웨어 프로젝트로서 공개 분산형 구성 서비스, 동기 서비스 및 대용량 분산 시스템을 위한 네이밍 레지스트리를 제공한다. 주키퍼는 하둡의 한 하위 프로젝트이었으나 지금은 독립적인 상위 프로젝트이다. 주키퍼의 아키텍처는 중복 서비스를 이용한 고가용성을 제공한다. 클라이언트는 주키퍼 마스터가 응답을 하지 않으면 다른 주키퍼 마스터에게 요청을 한다.

- ZooKeeper is a distributed co-ordination service to manage large set of hosts. Co-ordinating and managing a service in a distributed environment is a complicated process

- The ZooKeeper framework was originally built at “Yahoo!” for accessing their applications in an easy and robust manner. Later, Apache ZooKeeper became a standard for organized service used by Hadoop, HBase, and other distributed frameworks.

Zookeeper란 무엇인가?

- 분산 응용 프로그램이 실행되는 시스템 그룹을 클러스터(Cluster) 라고 하고 클러스터 에서 실행되는 각 시스템을 노드(Node)



- 분산 응용 프로그램의 장점
  - 신뢰성(Reliability), 확장성(Scalability), 투명(Transparency)
- 분산 응용 프로그램의 단점
  - 경합(Race condition), 교착상태(Deadlock), 불일치(Inconsistency)

Zookeeper가 하는 역할은?

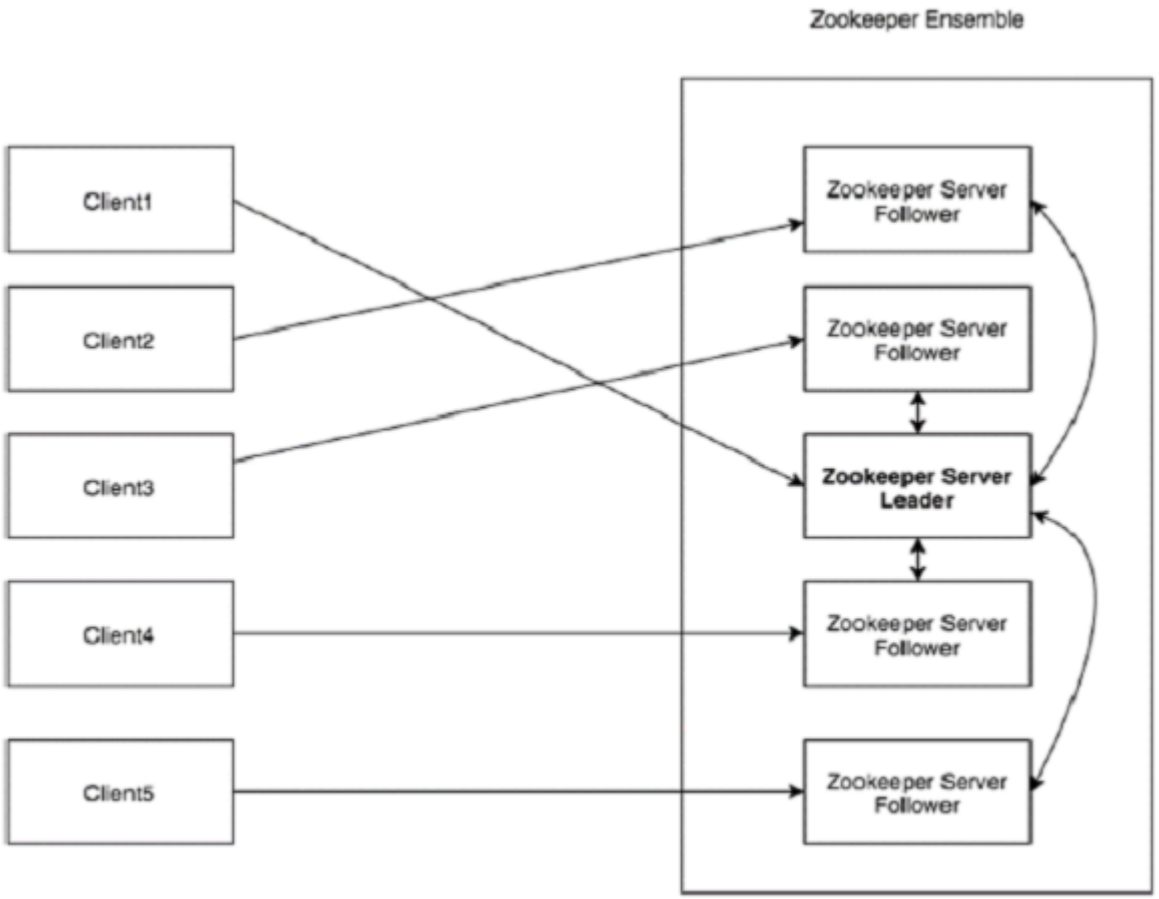
- Apache ZooKeeper는 클러스터 (노드 그룹)가 동기화 기술로 자체 데이터를 조정하고 공유 데이터를 유지 관리하는 데 사용함
  - Naming service
    - 클러스터의 노드를 식별하는 서비스, DNS서비스와 유사함
  - Configuration management
    - 노드들 간의 설정 정보를 동기화 해주는 해주는 기능
  - Cluster management
    - 노드의 등록/탈퇴 및 실시간 상태 체크
  - Leader election
    - Coordination 목적으로 노드의 Leader 선출기능
  - Locking & synchronization service
    - 데이터를 수정하면 Lock을 통해서 동기화 처리
  - Highly reliable data registry
    - 하나 또는 다수의 노드가 다운된 경우 데이터에 대한 가용성 제공

Zookeeper의 장점

- 단순한 분산 관리 프로세스
- 동기화
- 직렬화
- 신뢰성
- 원자성

Zookeeper Architecture

- Client-Server 구조를 가지고 있음

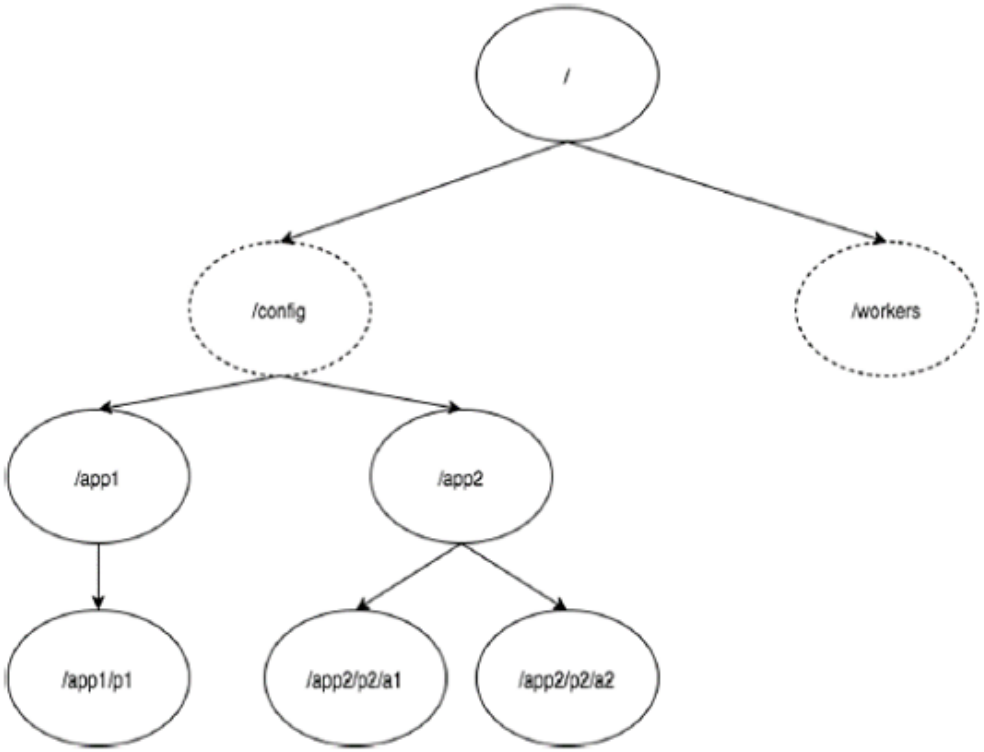


Zookeeper Components

- Client
  - 분산 응용 프로그램 클러스터의 노드 중 하나인 클라이언트는 서버의 정보에 액세스합니다. 특정 시간 간격 동안 모든 클라이언트는 서버에 메시지를 전송하여 클라이언트가 클라이언트가 살아 있음을 서버로 알립니다.
  - 클라이언트가 연결할 때 서버는 확인 응답을 보냅니다. 연결된 서버로부터 응답이 없으면 클라이언트는 자동으로 메시지를 다른 서버로 리다이렉션(redirect)합니다.
- Server
  - ZooKeeper 앙상블에있는 노드 중 하나 인 Server는 모든 서비스를 클라이언트에게 제공합니다. 서버가 작동중임을 알리기 위해 라이언트에게 알립니다.
- Ensemble
  - ZooKeeper 서버 그룹. 앙상블을 형성하는 데 필요한 최소 노드 수는 3입니다.
- Leader
  - 연결된 노드 중 하나라도 실패하면 자동 복구를 수행하는 서버 노드입니다. 리더는 서비스 시작시 선출됩니다.
- Follower
  - 선두 명령에 따르는 서버 노드.

Hierarchical Namespace

- 가장 첫번째 znode는 "/"로 구분됨
- 두가지 론리적인 namespaces를 가짐. config와 workers
- config namespace는 중앙으로 집중된 설정관리를 위해서 사용함
- worker namespace는 naming할때 사용함



* Zookeeper 데이터 모델

Hierarchical Namespace

- Zookeeper데이터 모델의 모든 znode는 stat구조를 가지고 있음
  - Stat는 단순한 znode의 metadata를 제공함

- Stat 구성요소
  - Version number
  - Action Control List(ACL)
  - Timestamp
  - Data length

- Znodes종류
  - Persistence znode
  - Ephemeral znode
  - Sequential znode

Sessions

- 세션은 ZooKeeper의 운영에 매우 중요합니다. 세션의 요청은 FIFO 순서로 실행됩니다. 클라이언트가 서버에 연결되면 세션이 설정되고 세션 ID 가 클라이언트에 할당됩니다.

- 클라이언트는 특정 시간 간격으로 하트 비트 를 전송 하여 세션을 유효하게 유지합니다. ZooKeeper Ensemble이 서비스 시작시 지정한 기간 (세션 시간 초과) 이상 동안 클라이언트로부터 하트 비트를 받지 못하면 클라이언트가 사망했다고 결정합니다.

- 세션 타임 아웃은 대개 밀리 초 단위로 표시됩니다. 어떤 이유로 세션이 종료되면 해당 세션 중에 작성된 임시 znode도 삭제됩니다.

Watchs

- 클라이언트가 ZooKeeper Ensemble의 변경 사항에 대한 알림을 받는 간단한 메커니즘입니다. 클라이언트는 특정 Znode를 읽는 동안 Watches를 설정할 수 있습니다. Watches 는 znode (클라이언트 레지스터에있는)가 변경된 경우 등록 된 클라이언트에 알림을 전송합니다.

- Znode 변경은 z 노드와 연관된 데이터의 수정 또는 z 노드의 하위 노드에서의 변경입니다. 시계는 한 번만 트리거(Trigger)됩니다. 클라이언트가 다시 알림을 원할 경우 다른 읽기 작업을 통해 완료해야합니다. 연결 세션이 만료되면 클라이언트와 서버의 연결이 끊어지며 연결된 Watches도 제거됩니다.

ZooKeeper Ensemble Nodes

- 단일 노드(a single node)가 해당 노드가 실패 할 경우, Zookeeper Ensemble가 실패합니다. "Single Point of Failure " 로 실제 운영환경에서는 권장되지 않습니다.

- 두 개의 노드(two nodes)중 하나의 노드에 오류가 발생이 하는 경우 다른 노트가 있으므로 장애없이 사용가능합니다.

- 세 개의 노드(three nodes)중 하나의 노드가 실패한 경우, ZooKeeper Ensemble 은 실제 프로덕션 환경에서 적어도 세 개의 노드를 가져야합니다.

- 네 개의 노드(four nodes)중 두 개의 노드가 실패한 경우, 다시 실패하면 세 개의 노드를 가진 상황과 유사하다. 여분의 노드는 어떤 용도로도 사용되지 않으므로 홀수의 노드를 추가하는 것이 좋습니다 (예 : 3, 5, 7).

Workflow

- Write
- Read
- Replicated Database
- Leader
- Follower
- Request Processor
- Atomic broadcasts