14장 유튜브 설계

1단계 문제 이해 및 설계 범위 확정

- 비디오 업로드, 재생이 제일 중요.
- 모바일 앱, 웹 브라우저, 스마트 TV 지원.
- 일간 사용자 5백만.
- 사용자는 평균적으로 30분 지원
- 다국어 지원.
- 현존하는 비디오 종류와 해상도 대부분 지원.
- 암호화 필요.
- 비디오 크기 최대 1GB.
- 벤더사의 클라우드 서비스 활용하도록 하면 좋음.

설계 초점

- 빠른 비디오 업로드
- 원활한 비디오 재생
- 재생 품질 선택 가능
- 낮은 인프라 비용
- 높은 가용성과 규모 확장성, 안정성
- 지원 클라이언트 : 모바일 앱, 웹브라우저, 스마트 TV

개략적 규모 추정

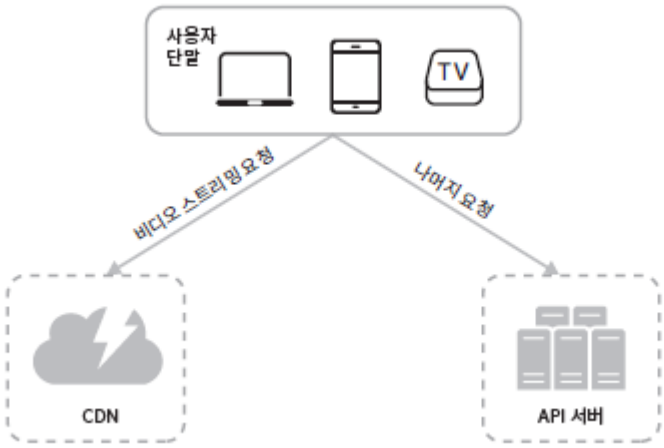
- 일간 능동 사용자 (DAU) : 5백만
- 한 사용자는 하루 평균 5개의 비디오 시청
- 10%의 사용자가 하루에 1비디오 업로드
- 비디오 평균 크기 300MB

- 비디오 저장을 위해 매일 새로 요구되는 저장 용량 = 5백만 x 10% x 300MB = 150TB
- CDN 비용
  - 클라우드 CDN을 통해 비디오를 서비스할 경우 CDN에서 나가는 데이터의 양에 따라 과금.
  - 아마존 CloudFront 를 CDN 솔루션으로 사용할 경우, 100% 트래픽이 미국에서 발생한다고 하면 1GB 당 $0.02의 요금이 발생한다. 문제 단순화를 비디오 스트리밍 비용만 따지자.
  - 따라서 매일 발생하는 요금은 5백만 X 5비디오 X 0.3GB X $0.02 = $150,000
  - 사업자가 큰 고객의 비용을 할인해준다고 해도 이렇게 하면 비용이 엄청나다. 따라서 그것을 줄이는 상세 설계가 필요하다.

2단계 개략적 설계안 제시 및 동의 구하기

이 설계에서는 CDN과 BLOB 스토리지의 경우에는 기존 클라우드 서비스를 이용할 것이다. 이유는 다음과 같다.

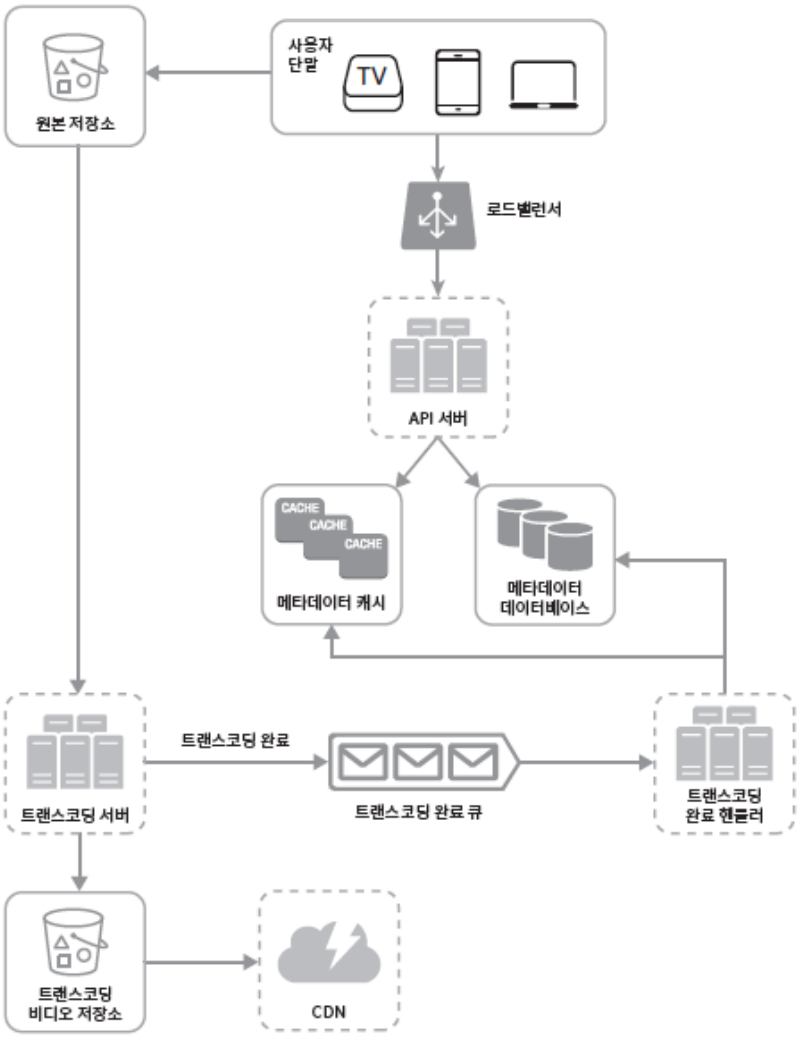
- 시스템 설계 면접은 모든 것을 밑바닥부터 만드는 것과는 관계 없다. 주어진 시간 안에 적절한 기술을 골라 설계를 마치는 것이, 그 기술 각각이 어떻게 동작하는지 상세히 설명하는 것보다 중요하다. 예를 들어, 비디오를 BLOB 저장소에 저장할거면 그 사실만 언급해도 된다.
- 규모 확장이 쉬운 BLOB 저장소나 CDN을 만드는 것은 복잡하고 비용이 많이 든다. 넷플릭스는 AWS를, 페이스북은 Akamai 의 CDN을 이용한다.
- 개략적으로 다음과 같이 구성된다.



- 단말 (client)

- CDN : 비디오는 CDN에 저장. 재생 버튼 누르면 CDN으로부터 스트리밍이 이루어진다.
- API 서버 : 비디오 스트리밍을 제외한 다른 요청을 처리.

비디오 업로드 절차



- 사용자
- 로드 밸런서 : API 서버 각각으로 고르게 요청을 분산
- API 서버 : 비디오 스트리밍 제외 모든 요청 처리
- 메타데이터 DB : 비디오의 메타데이터 보관
- 메타데이터 캐시 : 성능을 높이기 위해 비디오 메타데이터의 사용자 객체는 캐시.
- 원본 저장소 : 원본 비디오를 보관할 대형 이진 파일 저장소 시스템.

- 트랜스코딩 서버 : 비디오 인코딩이라고 부르기도 하는 것으로, 비디오의 포맷을 변환한다. 단말이나 대역폭 요구사항에 맞는 최적의 비디오 스트림을 제공하기 위해 필요하다.
- 트랜스코딩 비디오 저장소 : 트랜스코디잉 완료된 비디오를 저장하는 BLOB 저장소이다.
- CDN : 비디오를 캐시하는 역할을 한다. 사용자가 재생 버튼을 누르면 비디오를 송출한다.
- 트랜스코딩 완료 큐 : 트랜스코딩 완료 메시지 보관.
- 트랜스코딩 완료 핸들러 : 큐에서 이벤트를 꺼내어 메타데이터 캐시와 데이터베이스를 갱신할 작업 서버들.

그렇다면 비디오 업로드와 메타데이터 갱시이 병렬적으로 이루어진다고 한다면 어떤 프로세스로 진행될까?

프로세스 a : 비디오 업로드

1. 비디오를 원본 저장소에 업로드한다.
2. 트랜스 코딩 서버는 원본 저장소에서 해당 비디오를 가져와 트랜스코딩을 시작한다.
3. 트랜스코딩이 완료되면 아래 두 절차가 병렬적으로 수행된다.

   3a.1. 트랜스코딩이 끝난 비디오를 CDN에 올린다.

   3b.1. 완료 핸들러가 이벤트 데이터에를 큐에서 꺼낸다.

   3b.1.a, 3b.1.b 완료 핸들러가 메타데이터 데이터베이스와 캐시를 갱신한다.
4. API 서버가 단말에게 비디오 업로드가 끝나서 스트리밍 준비가 디었음을 알린다.

프로세스 b : 메타데이터 갱신

원본 저장소에 파일이 업로드 되는 동안, 단말은 병렬적으로 비디오 메타데이터 갱신 요청을 API 서버에 보낸다. 이 요청에 포함된 메타데이터에는 파일 이름, 크기, 포맷 드으이 정보가 들어있다.

비디오 스트리밍 절차

스트리밍 프로토콜

- 비디오 스트리밍을 위해 데이터를 전송할 때 쓰이는 표준화된 통신 방법
- 동작 원리를 정확히 애해하거나 이름을 외울 필요는 없다
- 프로토콜 마다 지원하는 비디오 인코딩과 플레이어가 다르므로 서비스에 맞는 거 선택해야함

- 비디오는 CDN에서 바로 스트리밍됨.

3단계 상세 설계

비디오 트랜스코딩

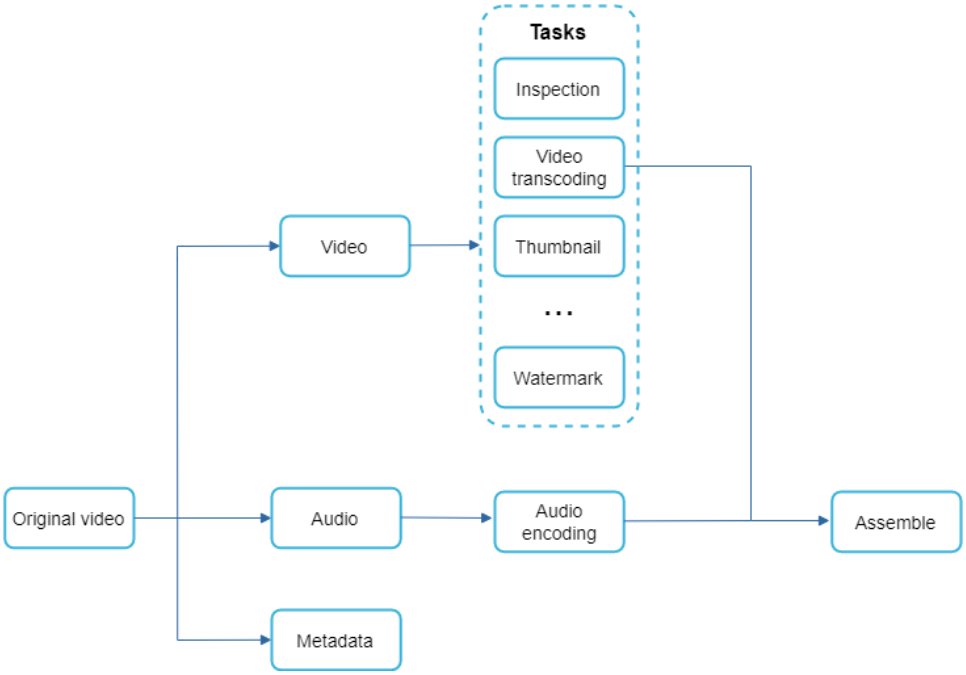
비디오를 녹화하면 단말은 해당 비디오를 측정 포맷으로 저장. 다른 단말에서도 순조롭게 재생되려면 다른 단말과 호환되는 bitrate와 포맷으로 저장되어야. bitrate는 비디오를 구성하는 비트가 얼마나 빨리 처리되어야 하는지 나타내는 단위로, 이게 높으면 고화질 비디오다.

비디오 트랜스코딩은 다음과 같은 이유로 중요하다.

- 가공되지 않은 원본 비디오는 저장 공간을 많이 차지한다.
- 상당수의 단말과 브라우저는 특정 종류의 비디오 포맷만 지원한다. 호환성을 위해 비디오를 여러 포맷으로 인코딩해 두는 게 좋다.
- 사용자에게 끊긴 없는 고화질 비디오를 재생하려면, 네트워크 대역폭 따라 저화질 고화질 비디오를 보내야 한다.
- 모바일 단말은 네트워크 상황이 수시로 달라질 수 있다. 비디오 끊김을 방지하려면 비디오 화질을 자동으로 혹은 수동으로 변경할 수 있도록 해야 한다.
- 인코딩 포맷 구성
    - 컨테이너 : 비디오 파일, 오디오, 메타데이터를 담는 바구니 같은 거. avi, mov, mp4 등 포맷.
    - 코덱 : 비디오 화질은 보존하고 파일 크기를 줄일 목적으로 고안된 압축 및 압축 해제 알고리즘. H.264, VP9, HEVC 등

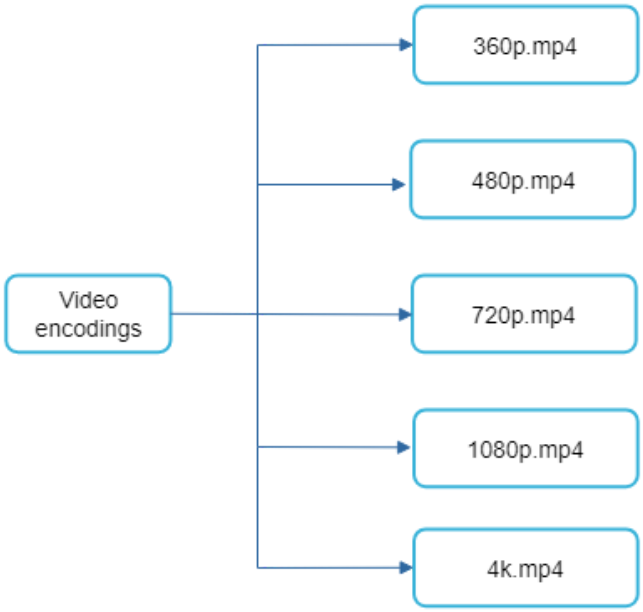
유향 비순환 그래프(DAG) 모델

비디오를 트랜스코딩 하는 자원 및 시간이 많이 소요된다. 워터마크나 섬네일 이미지 처럼 각자만의 요구사항을 가지기 때문이다. 이러한 다른 유형의 비디오 프로세싱을 지원하고 병렬성을 높이기 위해 실행 작업을 손수 정의하는 DAG를 도입하였다.



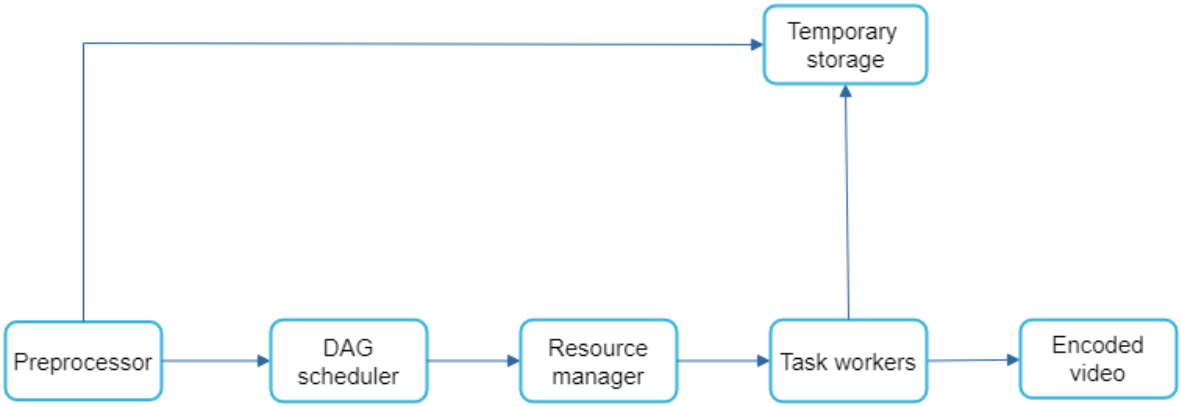
그림처럼 원본 비디오를 비디오, 오디오, 메타데이터의 세 부분으로 나누어 처리하였다. 비디오 처리를 자세히 살펴보자.

- 검사 (Inspection) : 좋은 품질의 비디오인지, 손상은 없는지 확인하는 작업
- 비디오 인코딩 : 비디오를 다양한 해상도, 코덱, 비트레이트 조합으로 인코딩 하는 작업이다.



- 섬네일 : 사용자가 업로드한 이미지나 비디오에서 자동 추출된 이미지를 이용.
- 워터마크 : 비디오에 대한 식별정보를 이미지 위에 오버레이 형태로 띄워 표시.

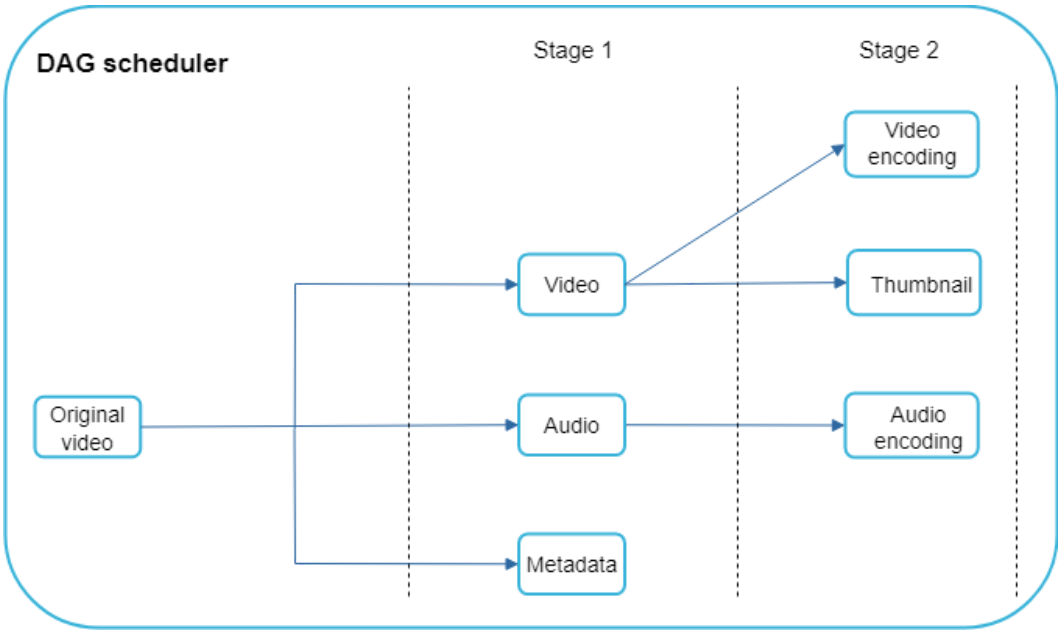
비디오 트랜스코딩 아키텍처



전처리기 : 다음 두 가지 일을 한다.

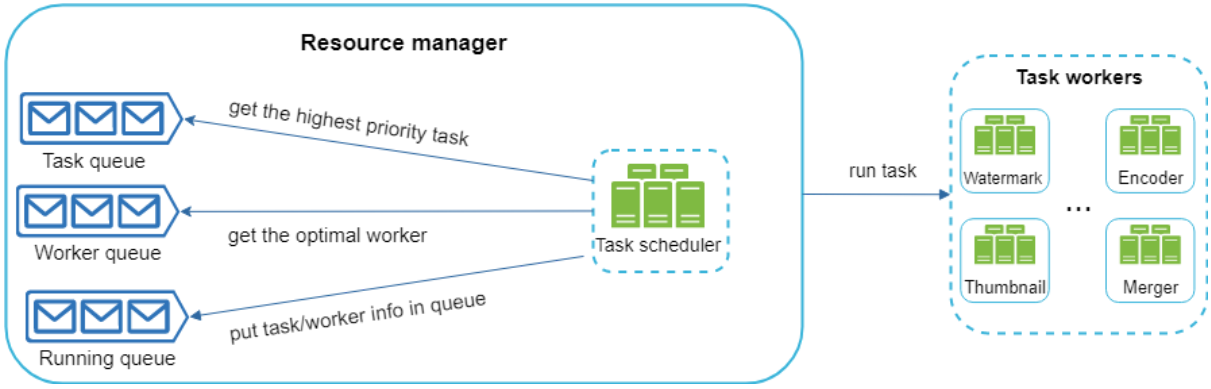
1. 비디오 분할 : 비디오 스트림을 GOP(Group of Pictures) 단위로 쪼갠다. GOP는 특정 순서로 배열된 프레임 그룹으로 독립적으로 재생 가능하고, 보통 몇 초 길이다.
2. DAG 생성 : 클라이언트 프로그래머가 작성한 설정 파일에 따라 DAG를 만들어낸다.
3. 데이터 캐시 : 전처리기는 분할된 비디오의 캐시이기도 하고, GOP와 메타데이터를 임시 저장소에 보관한다. 비디오 인코딩 실패시 시스템은 이런 보관된 데이터를 통해 인코딩을 계속한다.

DAG 스케줄러



DAG 그래프를 몇 단계로 분할하여 각각을 자원 관리자의 작업 큐에 넣는다. 그림은 하나의 DAG를 두 단계로 쪼갠 것이다.

자원 관리자



- 작업 큐 (task queue)
- 작업 서버 큐 (worker queue) : 작업 서버의 가용 상태 정보 보관.
- 실행 큐 (running queue) : 현재 실행 중인 작업 및 서버 정보 보관.
- 작업 스케줄러 : 최적의 작업/서버 골라서 수행 지시.
- 작업 스케줄러는 실행 지시 후 실행 큐에 정보를 넣은 뒤, 작업 완료 후 제거한다.

작업 서버

작업 종류에 딸 구분하여 관리.

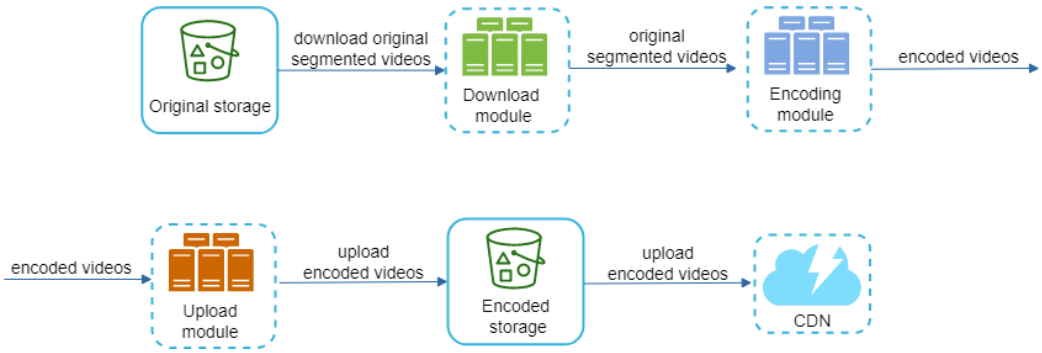
임시 저장소

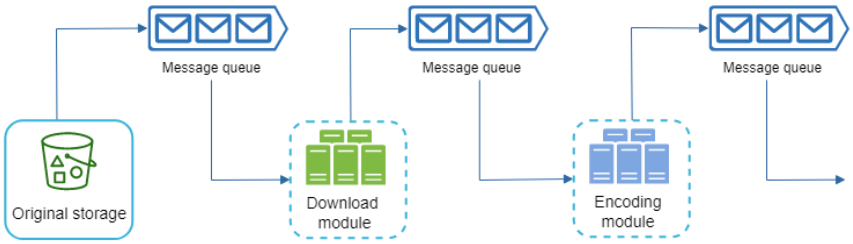
메타데이터는 작업 서버가 빈번히 참조하고 크기도 작으므로 메모리에 저장하는게 적합. 비디오/오디오는 BLOB 저장소에.

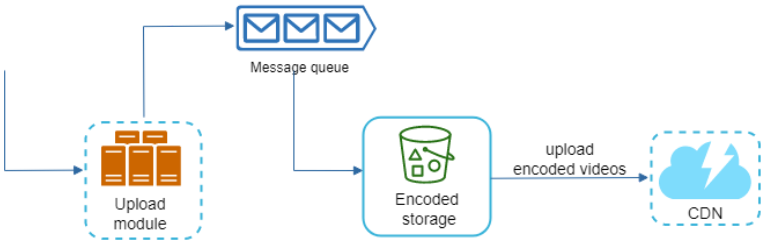
시스템 최적화

속도 최적화

- 비디오 병렬 업로드 : 작은 GOP로 분할 후 병렬적으로 업로드
- 업로드 센터를 사용자 근거리에 지정
- 모든 절차 병렬화 : 메시지 큐를 도입하여 이벤트 각각을 병렬적으로 처리한다.







안정성 최적화

- pre-signed upload URL : 허가받은 사용자만 올바른 장소에 업로드 할 수 있다. 이는 S3에서 사용하는 용어이고 Azure 의 BLOB 저장소에서는 Shared Access Signature 라고 한다.
- 비디오 보호
  - 디지털 저작권 관리 (DRM) 시스템 도입 : 애플의 FairPlay, 구글의 Widevine, MS 의 PlayReady
  - AES 암호화 (encryption) : 비디오를 암호화하고 접근권한 설정. 허락된 사용자만 재생시 복호화.
  - 워터마크

비용 최적화

유튜브의 비디오 스트리밍을 롱테일 분포를 따른다. 이를 이용하여 몇 가지를 시도해보자.

1. 인기 비디오는 CDN을 통해 재생하되 다른 비디오는 비디오 서버를 통해 재생.
2. 인기 없는 비디오는 인코딩할 필요도 없을 수도 있다. 짧은 비디오는 필요할 때 인코딩하여 재생.
3. CDN을 직접 구축하고 인터넷 사용 비용 낮추기 위해 ISP와 제휴한다.

오류 처리

- 회복 가능 오류 : 특정 비디오 세그먼트 트랜스코딩 실패 같은 오류. 몇 번 재시도 하면 해결. 그렇지 않다면 클라이언트에게 오류 코드 반환
- 회복 불가능 오류 : 비디오 포맷 오류 등. 작업 중단하고 오류 코드 반환.

전형적인 해결 방법

- 비디오 분할 오류 : 낡은 버전의 클라이언트가 GOP 따라 분할 못하는 경우. 서버에 전송하여 해당 비디오 처리하도록.
- 메타데이터 캐시 서버 장애 : 데이터는 다중화 되어 있으므로 다른 노드에서 데이터를 가져오도록. 장애난 서버는 교체.
- 메타데이터 데이터베이스 서버 장애
    - 주 서버 죽으면 부 서버 가운데 하나를 주 서버로 교체
    - 부 서버 죽으면 다른 부 서버 통해 읽기 연산 처리하고 주 서버는 교체