가상 면접 사례로 배우는

대규모 시스템 설계 기초

이정민

14. 유튜브 설계

참고

■ 유튜브 통계자료 (2020년 기준)

- 월간 능동 사용자 수 : 20억 명
- 매일 재생되는 비디오 수 : 50억 회
- 미국 성인 가운데 73%가 유튜브 이용 / 5천만 명의 창작자
- 유튜브의 광고 수입 : 2019년 기준 150억 달러 (2018년 대비 36% 증가)
- 모바일 인터넷 트래픽 가운데 37%를 유튜브가 점유
- 80개 언어로 이용 가능

1단계) 문제 이해 및 설계 범위 확정

- 요구사항 파악
  - 중요한 기능은? 비디오 업로드와 시청 기능 / 지원 클라이언트 : 모바일 앱, 웹 브라우저, 스타트 TV
  - 일간 능동 사용자 수? 5백만 명 / 사용자 평균 이용시간 : 30분
  - 다국어 지원? 필요함 / 지원 비디오 종류 및 해상도? 현존 비디오 종류 및 해상도 대부분
  - 암호화 지원? 필요함 / 비디오 파일 크기 제한? 최대 1GB로 제한
  - 클라우드 서비스 활용 가능? 가능함
- 요구사항 정리
  - 빠른 비디오 업로드, 원활한 비디오 재생, 재생 품질 선택 기능
  - 낮은 인프라 비용, 높은 가용성과 규모 확장성/안정성, 모바일 앱/웹브라우저/스마트TV 지원

1단계) 문제 이해 및 설계 범위 확정

■ 개략적 규모 추정

- 일간 능동 사용자 수 : 5백만 명
- 한 사용자는 하루에 평균 5개의 비디오를 시청
- 10% 사용자가 하루에 1 비디오 업로드
- 비디오 평균 크기는 300MB
- 비디오 저장을 위해 매일 새로 요구되는 저장 용량 : 150TB(5백만 * 10% * 300MB)
- CDN 일당 비용 : $150,000 (5백만 * 5비디오 * 0.3GB * $0.02, 아마존 클라우드 프론트 사용시)

2단계) 개략적 설계안 제시 및 동의 구하기

■ 주요 컴포넌트

- 단말 : 컴퓨터, 모바일 폰, 스마트 TV
- CDN : 비디오 저장. 재생 시 CDN으로 부터 스트리밍이 이루어 짐
- API 서버 : 비디오 스트리밍을 제외한 요청 처리
  - 피드 추천, 비디오 업로드 URL 생성, 메타데이터 데이터베이스 및 캐시 갱신, 사용자 가입 등

2단계) 개략적 설계안 제시 및 동의 구하기

■ 비디오 업로드 구성요소

- 사용자 / 로드밸런서 / API 서버
- 메타데이터 데이터베이스 : 비디오 메타데이터 보관, 샤딩/다중화 적용 (성능/가용성 요구사항 충족)
- 메타데이터 캐시 : 성능을 높이기 위해 비디오 메타데이터와 사용자 객체 캐시
- 원본 저장소 : 원본 비디오를 보관할 대형 이진 파일 저장소(BLOB)
- 트랜스코딩 서버 : 비디오 인코딩 역할. 비디오 포맷을 변환(단말/대역폭에 맞는 최적의 스트리밍)
- 트랜스코딩 비디오 저장소 : 트랜스코딩 완료된 비디오를 저장하는 BLOB 저장소
- CDN : 비디오 캐시 역할 / 트랜스코딩 완료 큐/핸들러 : 완료 이벤트 저장 및 캐시/DB 갱신 역할

2단계) 개략적 설계안 제시 및 동의 구하기

- 비디오 업로드 프로세스
  - 비디오 업로드 : p. 255 그림 14-5
  - 메타데이터 갱신 : p. 256 그림 14-6

2단계) 개략적 설계안 제시 및 동의 구하기

- 비디오 스트리밍 절차
  - 스트리밍 프로토콜
    - MPEG-DASH, Apple HLS, Microsoft Smooth Streaming, Adobe HDS
  - CDN에서 바로 스트리밍

3단계) 상세 설계

■ 비디오 트랜스코딩

- 다른 단말과 호환되는 비트레이트와 포맷으로 저장 (미가공 원본 비디오는 저장 공간이 많이 필요함)
- 상당수의 단말과 브라우저는 특정 종류의 비디오 포맷만 지원
- 사용자에게 고화질 비디오 재생을 보장하려면 비디오를 여러 포맷으로 인코딩하는 것이 바람직함
  - 네트워크 대역폭에 따라 저화질, 고화질 구분
- 모바일 단말의 경우 네트워크 상황이 수시로 변함. 비디오 화질을 자동/수동으로 변경할 필요 있음
- 인코딩 포맷
  - 컨테이너 : 비디오 파일, 오디오, 메타데이터를 담음(.avi, .mov, .mp4)
  - 코덱(codec) : 압축/해제 알고리즘 (H. 265, VP9, HEVC)

3단계) 상세 설계

■ 유향 비순환 그래프(DAG) 모델

- 작업을 단계별로 배열하여 작업들이 순차적 또는 병렬적으로 실행되도록 함
- 원본 비디오는 비디오, 오디오, 메타데이터로 나누어 처리 됨
- 비디오 부분에서 적용되는 작업
  - 검사 : 좋은 품질의 비디오인지, 손상이 없는 지 확인하는 작업
  - 비디오 인코딩 : 비디오를 다양한 해상도, 코덱, 비트레이트 조합으로 인코딩 함
  - 썸네일 : 사용자가 업로드한 이미지나 비디오에서 자동추출
  - 워터마크 : 비디오에 대한 식별정보를 이미지 위에 오버레이 형태로 표시

3단계) 상세 설계

- 비디오 트랜스코딩 아키텍처
  - 전처리기 : 비디오 분할, DAG 생성, 데이터 캐시
  - DAG 스케줄러 : DAG 그래프를 몇 개 단계로 분할하여 자원관리자의 작업큐에 푸시
  - 자원 관리자 : 자원 배분을 효과적으로 수행하는 역할
    - 작업 큐 : 실행 할 작업 보관 / 작업 서버 큐 : 작업서버의 가용상태 정보 보관
    - 실행 큐 : 현재 실행 중인 작업 및 작업 서버 정보 보관 / 작업 스케줄러 : 최적의 작업/서버 조합 선택, 작업 지시
  - 작업서버 : DAG에 정의된 작업 수행. 작업 종류에 따라 작업 서버 구분 관리
  - 임시저장소 : 비디오/오디오 데이터는 BLOB 저장소가 바람직함. 비디오 프로세싱 완료 시 삭제
  - 인코딩된 비디오 : 인코딩 파이프라인의 최종 결과물

3단계) 상세 설계

- 시스템 최적화
  - 속도 최적화
    - 비디오 병렬 업로드 : 분할된 작은 GOP 단위로 병력 업로드
    - 업로드 센터를 사용자 근거리에 지정 : 지리적으로 분산된 업로드 센터 (CDN을 업로드 센터로 이용)
    - 모든 절차를 병렬화 : 메시지 큐를 도입하여 시스템의 결합도를 느슨하게 낮춤 (파이프라인)
  - 안정성 최적화
    - 미리 사인된 업로드 URL : 허가 받은 사용자만 업로드 / 비디오 보호 : 디지털 저작권 관리, AES 암호화, 워터마크
  - 비용 최적화
    - 인기 비디오는 CDN을 통해 재생(그외는 비디오 서버), 인기 없는 경우 인코딩 생략. 짧은 비디오는 필요 할 때 인코딩하여 재생
    - 특정 지역에서만 인기 높은 비디오 관리 / CDN 직접 구축, ISP와 제휴

3단계) 상세 설계

■ 오류 처리

- 회복 가능 오류 : 특정 비디오 조각 트랜스코딩 실패 등. 재시도 시 해결오류 지속 시 오류코드 반환
- 회복 불가능 오류 : 비디오 포맷 오류 등. 비디오 작업 종료 후 오류 코드 반환
- 오류에 대한 전형적인 해결 방법
  - 업로드 오류 : 몇 회 재시도 / 비디오 분할 오류 : 클라이언트 오류 시 서버에서 처리하도록 함
  - 트랜스코딩 오류 : 재시도 / 전처리 오류 : DAG 그래프 재생성 / DAG 스케줄러 오류 : 작업을 재스케줄링
  - 자원 관리자 큐 장애 : 사본을 이용 / 작업서버 장애 : 다른 서버에서 해당 작업 재시도
  - API 서버 장애 : 다른 API 서버로 우회 / 메타데이터 캐시 장애 : 다른 서버로 우회. 장애 서버는 교체
  - 메타데이터 데이터베이스 서버장애 : 주서버 장애 시 부서버를 주서버로 전환, 부 서버 장애 시 교체

4단계) 마무리

■ 추가 논의사항

- API 계층의 규모 확장성 확보 방안 : 수평적 규모 확장
- 데이터베이스 계층의 규모 확장성 확보 방안 : 데이터베이스 다중화 및 샤딩
- 라이브 스트리밍 : 비디오 업로드, 인코딩, 스트리밍에서 비슷하나,
  - 라이브 스트리밍의 경우 응답지연이 더 낮아야 함. 스트리밍 프로토콜 선정에 유의 해야 함
  - 라이브 스트리밍의 경우 병렬화 필요성은 낮음. 작은 단위의 데이터를 실시간으로 빨리 처리해야 함
  - 라이브 스트리밍의 경우 오류 처리 방법으로 다르게 해야 함. 시간이 오래 걸리면 안 됨
- 비디오 삭제 : 저작권 위반, 선정성, 불법 관련 비디오는 내려야 함. 업로드 시 식별 및 사용자 신고