

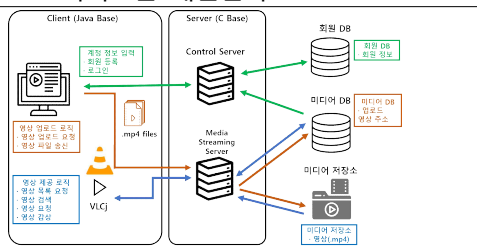
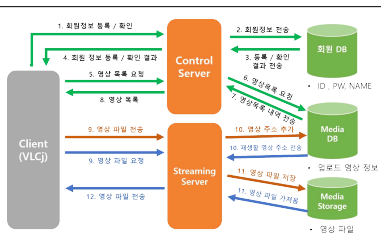
2019년도 1학기 네트워크 프로그래밍 최종보고서

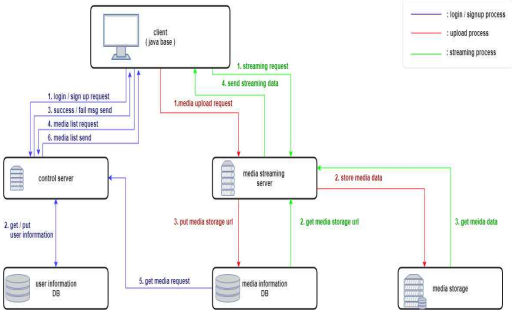
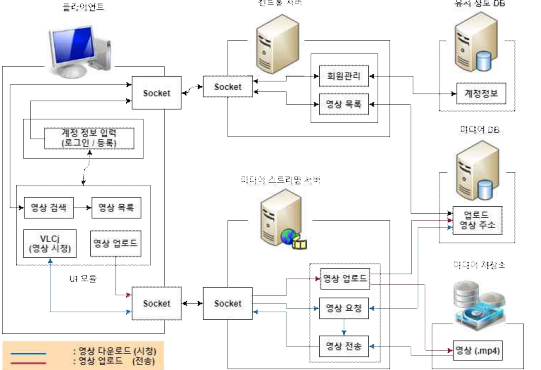
2019. 06. 20

| | |
|----------------|----------------------------------|
| 소 속 | 송실대학교 IT대학 컴퓨터학부 |
| 과 목 | 네트워크 프로그래밍 |
| 담 당 교 수 | 최 종 선 |
| 프로젝트 명 | SSUTube (실시간 ucc 스트리밍 서버) |

| <div>SSUTube</div> <div>progressive download 방식을 이용한 멀티 쓰레드 기반 동영상 스트리밍 서비스</div> | | | | |
|---|--|----------|---------|--------|
| 조편성 | 01조 | | | |
| 팀 명 | SSUGLE | | | |
| 팀 원 (총: 4명) | 이름 | 학번 | 개인별 역할 | 팀내 기여도 |
| | 계희준 | 20142300 | 스트리밍 서버 | 30 % |
| | 이상덕 | 20142408 | 컨트롤 서버 | 30 % |
| | 박성진 | 20142351 | DB | 20 % |
| | 강지현 | 20163341 | 클라이언트 | 20 % |
| 팀 구성 | <pre> graph TD TL[Team Leader : 계희준] --> IS[이상덕] TL --> PS[박성진] TL --> GJ[강지현] IS --> SS[스트리밍 서버] PS --> DB[DB] GJ --> CL[클라이언트] IS --> CS[컨트롤 서버] subgraph SS_Box [Streaming server] S1[영상 업로드 모듈] S2[영상 요청 모듈] S3[프로토콜 송수신 모듈] end subgraph CS_Box [Control server] C1[회원관리 모듈 (회원가입, 로그인)] C2[영상목록 요청 모듈] C3[Client 테스트 모듈] C4[프로토콜 송수신 모듈] end subgraph DB_Box [DB] D1[User info DB] D2[계정 정보 테이블 관리 모듈] D3[Media DB] D4[미디어 정보 테이블 관리 모듈] end subgraph CL_Box [Client] CL1[계정 정보 입력 모듈] CL2[영상 검색 모듈] CL3[영상 목록 출력 모듈] CL4[영상 재생 모듈] CL5[영상 업로드 모듈] CL6[프로토콜 송수신 모듈] end subgraph MS_Box [미디어 저장소] MS1[영상 전송 모듈] end </pre> | | | |

〈 개발계획서 요약 〉

| | | | | |
|----------------------------------|--------|---|--|-----------------|
| (조) 팀 명 | | (1조) SSUGLE | | |
| 프로젝트 명 | | SSU_tube - progressive download 방식을 이용한 멀티 쓰레드 기반 동영상 스트리밍 서비스 | | |
| 배경 및 당위성 | | 현재까지 송실대학교에서의 이벤트나 동아리 홍보는 게시판 위주의 정보 공유가 이뤄져왔다. 하지만 최근 스마트기기를 통한 영상 시청 인구가 많아지면서 텍스트 기반의 게시물을 통한 정보 공유뿐만 아니라 영상 매체를 통한 정보 공유가 활발해지고 있다. 따라서 송실대인이 서로 영상을 업로드하고, 동시에 시청할 수 있는 영상 스트리밍 서비스를 제공하고자 한다. | | |
| 제안 내용 | 최종 목표 | 본 프로젝트에서는 JVM환경에서 progressive download 방식으로 수신된 영상을 VLCJ를 사용하여 송실대 소속 학생들이 자유롭게 영상을 시청하고 업로드 할 수 있는 영상 스트리밍 SSU_tube 서비스를 개발한다. | | |
| | 시스템 개요 |  <p style="text-align: center;">그림 1 시스템 개요</p> |  <p style="text-align: center;">그림 2 통신 방식 개요</p> | |
| | 개발 방법 | <p>◎ client</p> <ul style="list-style-type: none">- 클라이언트는 서버에게 동영상 데이터 요청 및 업로드.- 클라이언트는 서버에게 회원가입, 로그인 요청과 영상 리스트를 요청.- 클라이언트는 개발된 플레이어 상에서 실시간으로 영상 데이터 수신 <p>◎ server</p> <ul style="list-style-type: none">- 서버에서는 여러 클라이언트의 요청을 동시에 처리.- 서버는 회원 관리(로그인, 회원가입) 와 동영상 관리(영상 데이터 송수신) 기능을 제공. <p>◎ DB</p> <ul style="list-style-type: none">- 회원 정보와 영상 데이터 정보를 DB에 저장. <p>◎ client</p> <ul style="list-style-type: none">- TCP를 이용하여 컨트롤 서버와 통신하여 로그인 및 회원가입 기능, 영상 리스트 요청.- TCP를 이용하여 스트리밍 서버에게 영상 데이터 요청 및 수신, 영상 업로드 요청.- VLCJ를 이용해 스트리밍 기능을 수행하는 플레이어 개발. <p>◎ server</p> <ul style="list-style-type: none">- C를 기반으로 한 멀티 쓰레드 서버를 구축.- 회원 관리와 동영상 관리를 각각 담당 하는 두 개의 프로세스 형태로 구축. <p>◎ DB</p> <ul style="list-style-type: none">- MySQL을 사용하여 구현. | | |
| 기대효과 (학습적 교육 효과 및 실용성 포함) | | <p>학습적 측면</p> <ul style="list-style-type: none">- 동영상을 출력하기 위해 VLCJ 프레임워크 사용법을 학습한다.- JAVA Swing을 사용해 클라이언트를 구축하는 과정을 통해 JAVA GUI프로그래밍을 학습한다.- 데이터 저장을 위해 MySQL서버를 사용하여 데이터베이스를 학습한다. <p>실용적 측면</p> <ul style="list-style-type: none">- 현 시점에서 송실대학교가 자체적으로 보유한 동영상 콘텐츠 제공 서비스가 존재하지 않다. 따라서 송실대학교의 자체적인 동영상 콘텐츠 제공 서비스를 구축할 수 있다.- 송실인들의 동아리 홍보 및 정보 공유가 더욱 활발하고 다양해 질 수 있다. | | |
| 중심어 | | Progressive download Java | VLCJ MySQL | media streaming |

| | |
|-------------|--|
| (조) 팀 명 | (1조) SSUGLE |
| 제목 | SSU_tube - progressive download 방식을 이용한 멀티 쓰레드 기반 동영상 스트리밍 서비스 |
| 시스템 구성도 | <div style="display: flex; justify-content: space-around;"> <div style="text-align: center;">  <p>그림 4 시스템 흐름도</p> </div> <div style="text-align: center;">  <p>그림 4 전체 시스템 구성도</p> </div> </div> |
| 주요 SW 모듈 | <ul style="list-style-type: none"> ○ 컨트롤 서버 <ul style="list-style-type: none"> - TCP 통신 모듈 : 컨트롤서버와 Client간의 로그인 및 회원가입 등의 회원관리 관련 통신을 담당, - DB 통신 모듈 : 계정 정보 DB와 연동하여 회원 정보 저장 및 확인. ○ 스트리밍 서버 <ul style="list-style-type: none"> - TCP 통신 모듈 : 클라이언트에게 요청받은 영상의 데이터를 송신하는 역할을 담당. 업로드 시 클라이언트로부터 받은 영상 정보를 저장소에 저장. - DB 통신 모듈 : 영상 정보 DB와 연동하여 영상 데이터가 저장된 URL 저장 및 확인. - 저장소 연동 모듈 : 동영상 데이터를 관리. ○ Client <ul style="list-style-type: none"> - TCP 통신 모듈 : 컨트롤 서버와 Client간의 회원관리, 영상시청, 업로드 관련 통신을 담당 - 동영상 재생 모듈 : VLC 프레임워크를 사용하여 전달받은 데이터를 VLC플레이어를 통해 영상데이터를 재생하는 기능 담당 - UI 조작 모듈 : 로그인, 회원가입, 동영상 요청 - UI 출력 모듈 : 동영상 리스트 출력 ○ DB <ul style="list-style-type: none"> - 계정 정보 DB: 유저들의 고유 정보를 저장하고 관리하는 기능을 담당 - 영상 정보 DB: 유저들이 업로드한 동영상 정보를 저장하고 관리하는 기능을 담당 |
| 위험요소 | <ul style="list-style-type: none"> - 스트리밍 서버 개발을 처음 접해보기 때문에 동영상 인코딩 처리에 어려움이 있다. - 영상 데이터 송수신의 패킷의 크기에 의해 서버에 부하가 생길 수 있다. - socket 서버 구축해본 경험자가 없어서 예외처리에 어려움이 있다. - 서버 저장소의 용량 한계가 있고 지원하지 않는 형식의 파일을 업로드 할 수 있다. |
| 위험요소의 해결 방법 | <ul style="list-style-type: none"> - 다양한 코덱을 지원하는 VLC player를 사용한다. - 회원관리와 스트리밍의 기능을 담당하는 두 개의 프로세스로 운용한다. - 가능한 많은 예외상황을 만들어 테스트해 본다. - 업로드 할 수 있는 영상의 크기 및 파일의 형식을 제한한다. |

목 차

| | |
|-------------------------------|----|
| 1. 프로젝트 개요 | 1 |
| 1-1. 개발 기술의 개요 | 1 |
| 1-2. 개발 기술의 필요성 | 1 |
| 1-3. 개발 기술의 예상효과 및 활용방안 | 1 |
| 2. 관련기술 현황 | 3 |
| 2-1. 관련기술 | 3 |
| 2-2. 요구분석 | 4 |
| 3. 수행내용 및 결과 | 5 |
| 3-1 개발 목표 | 5 |
| 3-2. 개발내용 | 5 |
| 3-2-1. 개발내용 및 범위 | 7 |
| 3-2-2. 구현내용 | 8 |
| 3-2-3. 개발결과 | 13 |
| 4. 장애요소 및 해결방법 | 14 |
| 5. 개발 후기 | 15 |

<그림 목차>

| | |
|---|----|
| [그림 1] 시스템 개요 | 1 |
| [그림 2] Progressive download 의 미디어 재생 방식 | 3 |
| [그림 3] 전체 시스템 구성도 | 7 |
| [그림 4] 서버 측 로그인 성공 프로토콜 패딩 후 전송 코드 | 9 |
| [그림 5] 클라이언트 측 로그인 요청 프로토콜 패딩 후 전송 코드 | 10 |
| [그림 6] 업로드_UI | 10 |
| [그림 7] 업로드 시 클라이언트와 서버의 consol | 10 |
| [그림 8] 영상 요청 시 서버 화면 | 10 |
| [그림 9] 스트리밍 영상 재생 화면 | 11 |
| [그림 10] user DB | 11 |
| [그림 11] media DB | 12 |

<표 목차>

| | |
|----------------------------|----|
| [표 1] 요구분석 | 4 |
| [표 2] 개발내용 | 6 |
| [표 3] 개발방법에 따른 실험 결과 | 13 |
| [표 4] 장애요소 및 해결방법 | 14 |
| [표 5] 에로사항 | 15 |
| [표 6] 개발후기 | 16 |

1. 프로젝트의 개요

1-1 개발 기술의 개요

본 프로젝트 SSUTube는 숭실인 들의 ucc 공유를 목적으로 하는 영상 공유 스트리밍 서비스이다.

SSUTube의 클라이언트는 숭실인들의 로그인 할 수 있는 로그인 화면과 로그인에 성공하면 동영상 재생 화면으로 구성된다. 또한 많은 숭실인들이 다양한 OS 환경에서 사용할 수 있도록 JAVA 기반으로 클라이언트를 구현하였다.

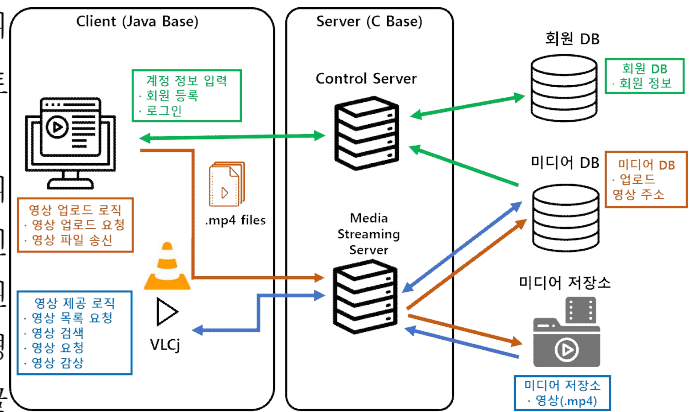


그림 1 시스템 개요

SSUTube의 서버는 각 서버의 부하를 막고 안정성을 높이기 위해 컨트롤 서버, 스트리밍 서버로 분리하여 구현하였다. 컨트롤 서버에서는 회원 가입 및 로그인, 동영상 검색기능을 수행한다. 스트리밍 서버에서는 클라이언트의 동영상 업로드 요청 및 동영상 재생 요청 시 동영상 데이터를 송수신 하는 역할을 한다.

DB는 회원 정보를 저장하는 DB와 동영상 데이터의 고유 키 값과 실제로 저장된 미디어 저장소의 URL 주소를 저장하고 있는 미디어 DB로 나누어 구현하였다.

1-2 개발 기술의 필요성

숭실대 학우들이 많이 사용하는 커뮤니티 어플리케이션으로는 에브리타임과 슈팅 등이 있다. 하지만 이러한 텍스트 기반의 어플리케이션은 영상기반의 어플리케이션 보다 직관적이지 못하고, 역동적인 홍보 효과가 부족하다. 그러므로 숭실인들의 동영상을 통한 정보교환 및 홍보를 위해 동영상 스트리밍 서비스인 SSUTube 플랫폼이 필요하다고 판단되어 개발을 진행하게 되었다.

1-3 개발 기술의 예상효과 및 활용방안

SSUTube는 숭실인들의, 숭실인들에 의한, 숭실인들을 위한 동영상데이터를 공유할 수 있는 플랫폼을 제공한다. 기존의 숭실인들의 데이터 공유 형태는 SNS, 에브리타임과 같은 타사의 플랫폼을 중심으로 이루어져 왔다. 이는 무분별한 정보의 홍수 속에서 신뢰성과 접근성을 낮추는 문제가 있다. 이에 비해 공식 어플리케이션인 슈팅의 경우 신뢰성과 접근성이 높

지만 제공하는 데이터가 텍스트로 한정되어, 영상데이터를 제공하지 않기 때문에 콘텐츠의 직관성이 영상에 비해 상대적으로 낮다는 단점이 있다. 이에 송실인들의 영상 공유 서비스인 SSUTube를 통해 우리는 독자적인 영상제공 플랫폼을 형성하고, 이전보다 직관적인 데이터를 배포할 수 있다.

SSUTube를 통해 송실인들의 동아리 및 행사 홍보영상부터 학교의 공식 영상, 더 나아가 송실인의 브이로그까지 영상기반의 모든 데이터를 공유할 수 있다. 기존의 슈팅을 통한 텍스트 기반 데이터에 영상기능을 추가함으로써 그 콘텐츠 제공의 한계를 벗어나 높은 활용도를 가진 독자적인 플랫폼을 구축할 수 있을 것이다.

2. 관련 기술 현황

2-1 관련 기술

2-1-1. youtube

SSUTube의 영상 스트리밍을 위해 youtube에서 사용하던 스트리밍 기법인 progressive download 기법을 사용하였다.

progressive download 방식은 미디어 파일을 서버에서 클라이언트로 전송할 때 클라이언트에서 데이터를 전부 받을 때까지 기다렸다가 재생하는 다운로드 방식이 아닌 데이터를 받으면서 파일을 재생하는 방식이다. 즉 서버에 저장된 파일을 내려 받으면서 동시에 파일을 재생한다.

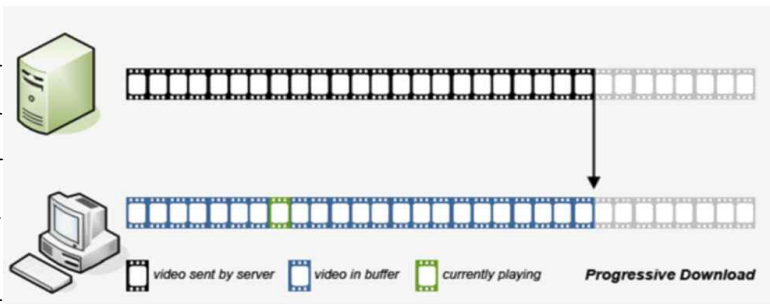


그림 2 Progressive download 의 미디어 재생 방식

progressive download 방식은 RTSP 방식과 달리 전송된 미디어 데이터가 클라이언트의 저장소에 저장된다. 동영상 파일의 헤더 정보와 같이 파일을 재생할 수 있는 일부 데이터가 전송이 되면 클라이언트에서는 미디어 파일을 재생한다. 이때 미디어 파일을 재생하며 계속해서 동영상 데이터를 서버로부터 전송 받는다. 따라서 동영상 데이터를 전부 받기 위해 기다리는 시간을 절약할 수 있다.

progressive download 방식은 RTSP에 비해 구현하기 쉽다는 장점이 있다. 하지만 클라이언트에 동영상 파일이 저장되는 방식이므로 보안 측면에서 문제가 발생할 수 있다. 또 유료 비디오 서비스의 경우 이 방식은 적절하지 않다. 게다가 미디어 데이터가 클라이언트에게 순서대로 도달해야 스트리밍 기능을 보장할 수 있다. 이와 같은 특성 때문에 동영상의 중간 부분을 보기 위해서는 앞선 데이터를 모두 전송받아야 한다. 또한 데이터의 일부만 재생하기를 원해도 전체 데이터를 다운 받아야 하는 경우가 생길 수 있다는 단점이 존재한다.

본 프로젝트는 숭실대 학생들의 UCC 공유를 목적으로 두고 있다. UCC는 동영상이지만 공유를 목적으로 한다. progressive download 방식의 단점에 큰 영향을 받지 않을 정도의 작고 짧은 영상이다. 따라서 본 프로젝트는 progressive download 방식을 활용하여 개발한다.

2-1-2. 커뮤니티 어플리케이션

숭실대 학우들이 많이 사용하는 커뮤니티 어플리케이션에는 대표적으로 슈팅과 에브리타임이 있다.

대학생들의 정보 공유 플랫폼인 에브리타임은 각 학교별로 독립된 게시판을 제공하는 커뮤니티 어플리케이션이다. 제공하는 데이터의 형태는 동영상을 제외한 이미지, 텍스트이다. 해당 대학에 재학중인 인증된 사용자만 게시글을 익명으로 업로드할 수 있다.

숭실대학교에서 공식적으로 제공하는 정보 공유 플랫폼인 슈팅은 각 단과대별로 독립된 게시판을 제공하는 커뮤니티 어플리케이션이다. 제공하는 데이터의 형태는 텍스트이다. 게시글을

업로드할 수 있는 대상은 학과 사무실로 신뢰성이 매우 높다는 장점이 있다.

따라서 본 프로젝트는 에브리타임의 장점인 접근성과 슈팅의 장점인 신뢰성을 바탕으로 자체적인 커뮤니티 어플리케이션인 SSUTube를 개발한다. 또한 위 두 어플리케이션의 텍스트 기반의 정보들의 한계점을 벗어난 영상 기반의 스트리밍 플랫폼을 개발한다.

2-2 요구 분석

[표 1] 요구분석

| | 슈팅 | 에브리타임 | YouTube | SSUTube |
|-----------|----------------|----------|---------------------|----------|
| 컨텐츠 제공 형태 | 텍스트 | 텍스트 | 동영상 | 동영상 |
| 이용자 수 | 송실대 학생 수 | 학교별 학생 수 | 약 3천 만 명 (1일 기준) | 송실대 학생 수 |
| 이용자 형태 | 송실대 학생 및 교원 | 해당 학교 학생 | 불특정 다수 | 송실인 |
| 신뢰성 | 매우 높음 | 매우 낮음 | 낮음 | 높음 |

슈팅과 에브리타임은 주요 컨텐츠가 텍스트이다. 텍스트는 영상에 비해 데이터양이 많지 않기 때문에 회원 정보를 다루는 서버와 데이터를 다루는 서버가 단일로 이루어져도 문제가 없다. 그러나 SSUTube 나 YouTube의 경우 영상을 공유하기 때문에 서버와 클라이언트 간 지속적으로 연결이 보장되어야 하므로 서버에 부하가 많이 가게 된다. 따라서 회원 관리는 컨트롤 서버가, 영상 업로드 및 재생은 스트리밍 서버로 나누어서 클라이언트와 각각 통신하도록 하였다. 이용자 수가 많은 YouTube의 경우, 구글의 데이터 센터를 사용한다. SSUTube의 경우는 송실대 생 내 DB에 저장된, 다시 말해 회원가입한 사람에 한해 이용할 수 있다. 따라서 이용자 수가 YouTube에 비해 다소 적은 편이므로 일반 PC를 서버 컴퓨터로 사용한다.

YouTube는 자료를 업로드가 할 수 있는 대상이 불특정 다수이고, 가입 후에 업로드가 가능하더라도, 업로더에 대한 정보를 알기 어려우므로 데이터에 대한 신뢰성이 높지 않다. 에브리타임 또한 익명성이 보장되므로, 역시 데이터에 대한 신뢰성이 높지 않다. 이에 비해 슈팅은 교내 시스템에서 인증된 사용자만 로그인 후 데이터를 올릴 수 있으므로 데이터에 대해 신뢰성이 매우 높은 편이다. SSUTube는 DB에 저장된 사용자만 로그인하여 업로드할 수 있으므로 YouTube나 에브리타임에 비해 신뢰도가 높다.

3. 수행내용 및 결과

3-1 개발 목표

본 프로젝트인 SSUTube는 Progressive Download 방식을 바탕으로 송실인들의, 송실인들에 의한, 송실인들을 위한 멀티쓰레드 기반의 독자적인 동영상 데이터 제공 플랫폼을 개발한다.

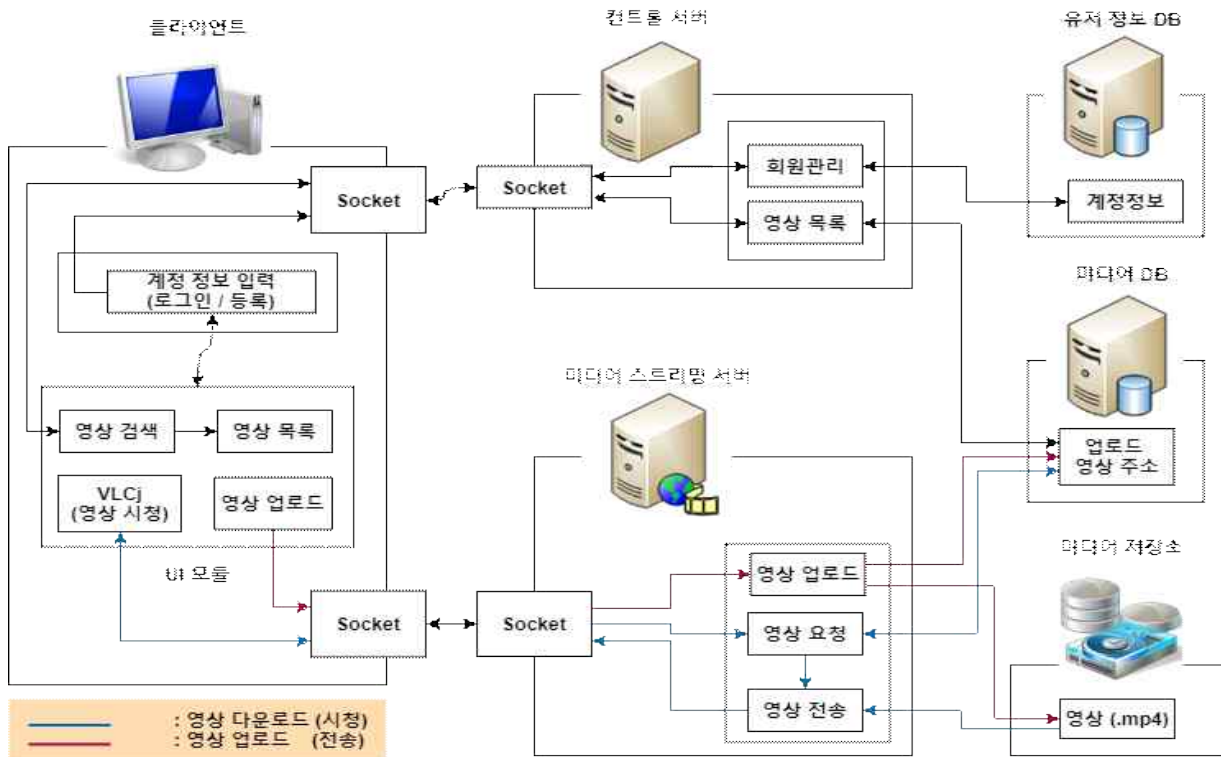
3-2 개발내용

[표 2] 개발내용

| 개발목적 | 연구개발 내용 | 성과내용 |
|--------------|---|---|
| Client 구현 | <ul style="list-style-type: none"> • 회원가입 및 로그인 기능 • 영상 업로드 • 영상 스트리밍 • 영상 검색 | <ul style="list-style-type: none"> • 자바기반의 GUI를 통해 각 버튼을 통해 서버와 통신할 수 있는 핸들러를 구현 • 쓰레드를 이용해 영상 데이터를 서버에게 전송 • VLC 프레임워크를 이용하여 파일 포맷에 구애받지 않고 영상을 스트리밍으로 재생 • ID와 Title에 따라 검색 키워드를 서버에 요청할 수 있는 핸들러를 구현 |
| Server 구현 | <ul style="list-style-type: none"> • 회원 정보 관리 • 영상 저장 | <ul style="list-style-type: none"> • DB와 통신하여 회원가입, 로그인 등의 기능을 수행. 자바로 구현된 클라이언트와 통신을 위해 Bytes가 아닌 string으로 통신 • 영상 저장 공간에 전송받은 영상데이터를 저장하고 DB를 갱신 |

| | | |
|-------|--|---|
| | <ul style="list-style-type: none"> • 영상 전송 • 영상 목록 추출 | <ul style="list-style-type: none"> • 키 값에 따른 DB에 저장된 Url을 참조하여 영상데이터를 클라이언트에 전송 • 클라이언트의 검색 요청에 따라 적절한 영상 목록을 DB에서 추출하여 클라이언트에 송신 |
| DB 구현 | <ul style="list-style-type: none"> • 회원 정보 저장 • 영상 정보 저장 | <ul style="list-style-type: none"> • 서버와 통신하여 MySQL-C API를 이용하여 적절한 쿼리문을 통해 회원 정보 DB 저장 및 중복확인 • 영상 고유의 키 값을 저장하여 DB에 저장 |

3-2-1 개발 내용 및 범위



[그림 3] 전체 시스템 구성도

개발 내용의 범위는 중간발표 때 제안했던 내용과 거의 동일하되 컨트롤 서버와 스트리밍 서버간의 소켓 통신을 배제하였다. 전체 구조는 위 그림처럼 3-Tier 형태로 구성되어 있다.

클라이언트는 VLCJ 프레임워크를 사용하고 다양한 환경에서 사용이 가능하도록 자바 환경으로 개발하였다. 또한 JAVA Swing을 이용하여 UI를 구성하였다.

서버는 회원 정보를 총괄하는 컨트롤 서버와 미디어 데이터 송수신을 전담하는 스트리밍 서버로 구성되어있다. 각 서버는 다중 사용자를 처리하기 위해 멀티쓰레드를 사용하여 구현하였다. DB는 C언어로 구축되어 있는 서버와 통신하기 위해 제공된 API를 통해 서버와 통신하고, 회원 정보를 관리하는 DB, 미디어 정보를 관리하는 DB로 구성하였다.

3-2-2 구현 내용

3-2-2-1 메시지 프로토콜

- 메시지 프로토콜은 첫 1~2 바이트를 통해 그 기능을 식별한다.

Client (Send) Control Server (Receive)

| | | | | |
|---|---------------|------------------|---------------|----------------|
| 1 | ID (10 Bytes) | | PW (10 Bytes) | |
| 2 | ID (10 Bytes) | PW (10 Bytes) | | Name (8 bytes) |
| 3 | | | | |
| 6 | 1 | ID (10 Bytes) | | |
| 6 | 2 | Title (60 Bytes) | | |

- 클라이언트에서 컨트롤 서버에 송신하는 프로토콜이다.

- 첫 바이트를 통해 1(로그인), 2(회원가입), 3(조회수 상위 10개 동영상 리스트 요청), 6-1(업로더로 검색), 6-2(이름으로 검색)을 구분한다.

Control Server (Send) Client (Receive)

| | | | | | | |
|---|---|------------------|---------------------|------------------|-------------------|--------------------|
| 1 | 1 | Name (8 Bytes) | | | | |
| | | | | | | |
| 1 | 2 | | | | | |
| | | | | | | |
| 2 | 1 | | | | | |
| | | | | | | |
| 2 | 2 | | | | | |
| | | | | | | |
| 3 | 1 | Count (4 Bytes) | | | | |
| | | | | | | |
| 3 | 2 | Key (4 Bytes) | Title (60 Bytes) | ID (10 Bytes) | SIZE (8 Bytes) | Views (4 Bytes) |

- 컨트롤 서버에서 클라이언트로 송신하는 프로토콜이다.

- 두 바이트를 통해 1-1(로그인 성공), 1-2(로그인 실패), 2-1(회원가입 성공), 2-2(회원가입 실패), 3-1(영상 데이터 개수 송신), 3-2(영상데이터 송신)을 구분한다.

Client (Send) Streaming Server (Receive)

| | | | |
|-------------------------|------------------|----------------|---------------|
| 4 | Title (60 Bytes) | Size (4 Bytes) | ID (10 Bytes) |
| Video Data (1024 Bytes) | | | |

| | |
|---|---------------|
| 5 | key (4 Bytes) |
|---|---------------|

- 클라이언트에서 스트리밍 서버로 송신하는 프로토콜이다.
- 첫 바이트를 통해 4(업로드 영상 정보), 5(영상 재생)을 구분한다.
- 영상데이터는 파일에서 1024바이트를 읽고 해당 데이터를 송신한다.
- 영상 재생은 미리 받은 key값을 이용하여 요청한다.

Streaming Server (Send) Client (Receive)

| |
|----------------|
| Size (4 Bytes) |
|----------------|

| |
|-------------------------|
| Video Data (1024 Bytes) |
|-------------------------|

- 스트리밍 서버에서 클라이언트로 송신하는 프로토콜이다.
- 클라이언트로부터 5번 프로토콜을 통해 영상의 재생을 요청받았을 때 최초 4바이트로 영상의 크기를 전송하고, 그 후 파일에서 1024바이트를 읽어 클라이언트에게 송신한다.

3-2-2-2 Control Server

클라이언트로부터 로그인, 회원가입, 미디어 데이터 요청과 같은 기능을 수신하고, 결과데이터를 클라이언트로 송신하는 서버이다.

○ 멀티쓰레드

해당 Control Server는 동시에 여러 사용자의 입출력을 처리하기 위해 멀티쓰레드 방식을 사용하였다. 각각의 클라이언트는 접속 시 하나의 소켓을 할당받고, 해당 소켓을 통해 데이터를 송수신한다. 클라이언트의 접속 종료시 서버에 소켓을 반납하게 되는데 이를 처리하면서 동기화 문제가 발생할 수 있기 때문에 Mutex를 통해 해당 문제를 해결하였다.

○ 프로토콜 파싱

JAVA로 구축한 Client와 C로 구축한 Server간의 프로토콜을 송수신하기 위해 문자열을 사용하였다. 기존에 JAVA의 데이터 저장 형태는 Big Endian으로 C의 Little Endian과 큰 차이가 있다. 따라서 클라이언트에서 정수형태의 데이터를 Byte배열로 송신할 시, 서버에서 정상적으로 읽지 못하였다. 하지만 문자열 데이터를 송수신하는 것은 아무런 제한사항이 없고, 데이터의 원본의 순서와 데이터가 그대로 유지된다는 점을 파악하여, [그림 X]와 같이 데이터를 송신할 때 '0'을 패딩으로 추가하여 클라이언트와 서버간의 통신을 할 수 있도록 하였다.

```
memset(msg, 0, 100);
strcpy(msg, LOGIN_SUCCESS);
strncat(msg, "00000000", 8-strlen(ct.name));
strcat(msg, ct.name);
send_msg(msg, 10, clnt_sock);
```

[그림 4] 서버 측 로그인 성공 프로토콜 패딩 후 전송 코드

따라서 클라이언트 측에서도 다음과 같은 방식을 통해 문자열을 생성, 서버로 송신해야 한다. [그림 X]는 클라이언트측 송신 코드이다.

```
String i = "1";
String ID = "ASDFA";
String PW = "CCCCC";
po.println(i + "000000000".substring(ID.length()) + ID + "000000000".substring(PW.length()) + PW);
```

[그림 5] 클라이언트 측 로그인 요청 프로토콜 패딩 후 전송 코드

해당 방식을 통해 서버와 클라이언트 간 정상적인 프로토콜 송수신을 할 수 있었다.

3-2-2-3 Streaming Server

○ 영상 데이터 업로드

클라이언트에서 영상 업로드 버튼을 클릭하게 되면 오른쪽 사진과 같이 영상을 선택 할 수 있는 화면이 보여 진다. 클라이언트가 영상을 선택하면 서버로 영상 데이터를 전송한다. 이때 영상의 크기를 먼저 서버로 보낸다, 업로드 요청을 받은 서버는 해당 크기만큼 데이터를 수신하고 upload 디렉토리 하위에 파일을 생성하고 저장한다. 파일을 생성할 때 파일이름 뒤에 업로드 시간 정보를 첨부하여 잘못된 요청으로 파일이 변경 되는 것을 방지한다. 파일 전송이 끝나면 DB에 URL 정보와 올린 사람 정보를 함께 저장하도록 요청한다. DB에 저장할 때 해당 영상이 고유 key 값을 가지도록 저장한다.

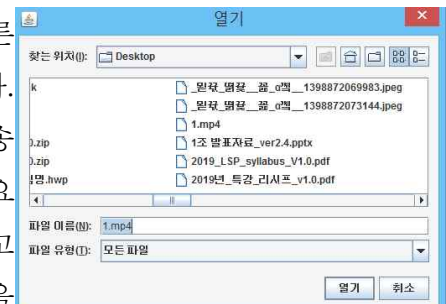


그림 [6] 업로드_UI

못된 요청으로 파일이 변경 되는 것을 방지한다. 파일 전송이 끝나면 DB에 URL 정보와 올린 사람 정보를 함께 저장하도록 요청한다. DB에 저장할 때 해당 영상이 고유 key 값을 가지도록 저장한다.

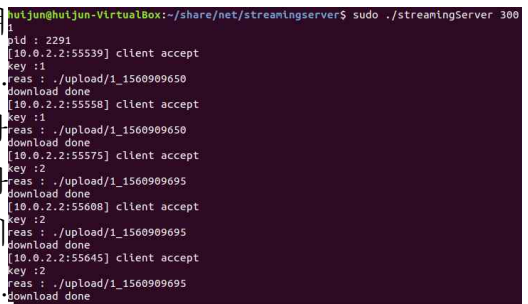
```
upload file size : 63170528 [10.0.2.2:56799] client accept
size : 63170528 length : 4 upload done
[B057be6bc9
done
```

[그림 7] 업로드 시 클라이언트와 서버의 consol

○ 영상 데이터 다운로드

클라이언트가 미리 전송 받은 key값을 이용해 서버에게 동영상을 요청한다. 서버는 요청을 받으면 미디어 데이터를 관리하는 DB와 통신하여 key값에 해당하는 동영상이 저장된

URL을 얻는다. 해당 URL을 open() 함수를 이용하여 열고 lseek() 함수를 이용해 파일의 크기를 얻는다. 동영상 데이터를 전송하기 전 4bytes 데이터를 클라이언트에게 보낸다. 클라이언트는 이 데이터를 받아 자바의 메모리 저장 방법에 맞추어 변환하고 해당 데이터만큼 미디어 파일 데이터를 서버로부터 받는다.

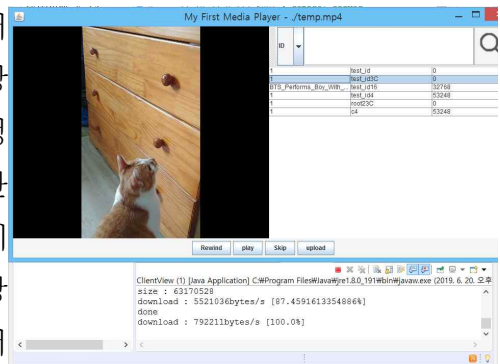


전송 할 데이터의 크기를 보낸 서버는 미디어 데이터 [그림 8] 영상 요청 시 서버 화면을 전송하기 시작한다. 일정 버퍼 크기만큼 파일에서 읽어 읽은 데이터를 바로 클라이언트에게 전송한다.

3-2-2-4 Media Play

○ 영상 데이터 실시간 재생

서버로 동영상을 요청하기 전 클라이언트는 먼저 키 값에 포함된 동영상 목록을 받아야 한다. 동영상 목록은 처음에 로그인 하게 되면 조회 수 순으로 정렬된 데이터를 받은 후 파싱하여 배열 형태로 저장한다. 그 후 jTable를 이용하여 사용자에게 동영상 제목과 올린 사람, 조회 수를 보여준다. 사용자가 해당 테이블을 클릭하게 되면 클릭한 동영상의 키 값을 서버로 보내게 된다. 서버는 키 값을 이용해 저장소에



[그림 9] 스트리밍 영상 재생 화면
서 동영상 데이터를 읽어 클라이언트로 전송한다. 클라이언트는 전송 받은 동영상 데이터를 저장 하면서 VLCJ 프레임워크를 이용하여 동영상을 재생한다. 이때 진행 상황을 eclipse console에 출력한다.

3-2-2-5 User Manage

```
mysql> select * from user;
```

| _id | _pw | _name |
|-------|----------|-------|
| a | b | c |
| duck | duck | duck |
| kye | 20142300 | key |
| root | leaf | groot |
| root2 | root2 | root2 |
| s | s | s |

8 rows in set (0.00 sec)

서버가 C언어로 구축 되었기 때문에 MySQL에서 제공하는 C API를 이용하여 DB와 서버간의 통신을 구현하였다. mysql_init() 함수를 통해 mysql 객체를 초기화 해준다. 이후 mysql_real_connect()를 통해 해당 DB와 연결 하고, 쿼리문 전달에는 mysql_query() 함수 와 mysql_real_query()를 이용하여 쿼리문을 전송하여 원하는 기능을 수행하였다.

[그림 10] user DB

○ 로그인

회원가입 기능은 위의 함수를 통해 DB에 접근하여 회원 관리 테이블에 INSERT 쿼리를 전달하여 회원을 DB에 추가한다. 회원가입을 할때는 중복을 막기위해 회원의 id 를 primary key로 설정하였다. 회원가입에 실패하는 경우는 같은 id가 DB에 입력되는 경우이다. 이에 대한 처리는 현재 테이블에 저장되어있는 row 수를 구하여놓고 INSERT IGNORE 쿼리를 통해 쿼리를 전달한 후의 row값을 비교하여 같다면 로그인 실패로 처리하였다. 서버에 로그인 성공시 1을 리턴해주었고, 실패시 0을 리턴해주었다.

○ 회원가입

로그인의 경우도 회원가입 처리과정과 거의 같다. 로그인에서는 SELECT 문의 추가 옵션을 통해 구현하였다. WHERE ~ AND ~ 쿼리를 통해 전달받은 회원 구조체에 대한 id 와 password에 대해 DB에서 검색하여 그에 맞는 row 를 체크하여 로그인 성공, 실패를 확인하였다. SELECT 문의 결과에 대해 mysql_num_rows()함수를 통해 row의 개수가 0개일 때 로그인 실패, 1일때 로그인 성공을 판단하였다. 로그인 성공 시 서버에 1을 리턴함과 동시에 call-by-reference를 통해 DB에 저장된 사용자의 이름을 전달하였다. 로그인 실패시 서버에 0을 리턴해주었다.

3-2-2-6 Media File Manage

○ 영상 검색

영상 검색은 업로드한 유저의 id를 통한 방법과 영상 제목인 title을 통한 방법 두 가지로 구현하였다. 구현한 함수의 첫번째 인자는 검색어가 전달 된다. 두번째 인자로는 검색 결과에 맞는 영상 데이터 구조체 배열을 가리키는 포인터이다. 이에 call-by-reference를 통해 서버에 검색 결과를 전달한다. 검색시 DB에 전달하는 쿼리는 SELECT ~ WHERE ~ LIKE 문을 이용하였다. 옵션을 통해 검색어가 포함된 row를 검색할 수 있도록 구현하였다. 이후 mysql_num_rows()함수를 통해 row의 개수를 구하여 구조체 배열의 수를 동적할당 하여 검색결과를 서버에 전달하였다.

○ 영상 정보 관리

클라이언트가 동영상 요청을 할 때 메시지 프로토콜의 크기를 줄이고 서버에서 보다 쉽게 영상을 찾기 위해 각 동영상 마다 고유한 key 값을 할당한다. 이 값은 DB에 저장되고 Primary Key와 AUTO_INCREMENT 옵션을 적용하여 구현하였다.

```
mysql> mysql> select * from media;
```

| _key | _title | _id | _size | _view | _url |
|------|---------------------------|---------|----------|-------|---|
| 1 | 1 | test_id | 0 | 0 | ./upload/1_1560909650 |
| 2 | 1 | test_id | 63170528 | 0 | ./upload/1_1560909695 |
| 3 | BTS_Performs_Boy_With_Luv | test_id | 23885960 | 0 | ./upload/BTS_Performs_Boy_With_Luv_1560921504 |
| 4 | 1 | test_id | 4476141 | 0 | ./upload/1_1560925094 |
| 5 | 1 | root2 | 63170528 | 0 | ./upload/1_1560925621 |
| 6 | 1 | c | 4476141 | 0 | ./upload/1_1560928077 |
| 7 | 1 | root2 | 63170528 | 0 | ./upload/1_1561025482 |

7 rows in set (0.00 sec)

[그림 11] media DB

3-2-3 개발결과

[표 2] 개발방법에 따른 실험 결과

| 구 분 | | 개발방법 | 개발 결과 |
|-------------|--------------|---|---|
| DB | 계정 정보 테이블 관리 | 서버에서 클라이언트로부터 받은 계정 정보를 튜플로 추가하고, id 및 pw에 해당하는 튜플을 추출한다. | id, title, name을 테이블 내에 저장하고, 서버에서 받은 값에 해당하는 id, pw가 있을 경우 결과 성공 전송 |
| | 영상 정보 테이블 관리 | 서버에서 클라이언트로부터 받은 영상 정보를 튜플로 추가하고, id 또는 title에 해당하는 튜플을 추출한다. | id, title, views, 영상 저장위치, 영상 크기를 테이블 내에 저장하고, 서버에서 받은 값에 해당하는 id, 또는 title에 속하는 튜플을 추출 |
| 컨트롤 server | 회원 관리 | 클라이언트로부터 받은 메시지 프로토콜에 따라 회원DB에 맞는 데이터를 요청, 수신한다. | 프로토콜에 따라 id, pw, name등을 미디어 DB로 전송하고 결과 값을 받아 클라이언트로 전송 |
| | 영상목록 요청 | 클라이언트로부터 받은 메시지 프로토콜에 따라 미디어 DB에 맞는 데이터를 요청, 수신한다. | 프로토콜에 따라 id 또는 title을 미디어 DB로 전송하고 결과 값을 받아 클라이언트로 전송 |
| | 프로토콜 송수신 | java로 작성된 클라이언트로부터 메시지 수신시 엔디안을 변경해 수신한다. | 프로토콜에 따라 문자열 생성 모듈을 작성하여 해당 문자열을 송수신 |
| 스트리밍 server | 영상 업로드 | 클라이언트로 받은 영상정보 및 저장된 위치를 미디어DB에 저장한다. | 미디어DB에 영상 저장위치 (/upload/)에 시간 정보를 포함하여 저장 |
| | 프로토콜 송수신 | java로 작성된 클라이언트로부터 메시지 수신시 엔디안을 변경해 수신한다. | 빅 엔디안을 리틀 엔디안으로 바꾸는 함수를 작성해 값을 수신 |
| | 영상 전송 | 클라이언트로 부터 받은 영상 키 값에 해당하는 영상이 저장된 주소를 DB로부터 가져와 일정한 크기로 클라이언트에 전송한다. | 키 값에 해당하는 정보를 미디어 DB에서 검색한 후 저장된 url을 이용해 저장된 파일에 접근하고 읽어서 클라이언트에 전송 |
| Client | 계정정보 입력 | 사용자가 id, pw를 또는 id, pw, name 정보를 입력 시 서버로 전송하게 한다. | id, pw, name을 프로토콜 내 지정된 크기만큼 0으로 채운 String의 substring으로 설정해 전송함 |
| | 영상 검색 | 사용자가 검색창에 id 또는 title에 해당하는 내용을 입력 시 서버로 전송하게 한다. | id, pw, name을 프로토콜 내 지정된 크기만큼 0으로 채운 String의 substring으로 설정해 전송함 |
| | 영상목록 출력 | 요청한 내용의 영상 목록을 서버로부터 받아 table 형태로 출력한다. | 서버로부터 받은 메시지를 프로토콜에 맞게 추출해 instance로 저장하고, id, title, views를 열로 갖는 table에 출력함 |
| | 영상 재생 | 영상 목록 table 내 목록을 더블클릭한 뒤, play버튼으로 재생한다. | 서버로부터 영상 데이터를 받아 tmp.mp4 파일로 저장하고, play버튼을 누르면 해당 파일을 재생 |
| | 영상 업로드 | upload버튼을 누르면 파일탐색기창이 뜨고 사용자가 mp4파일을 선택해 전송한다. C언어로 작성된 서버에 맞게 프로토콜을 작성해 전송하고, String또는 byte 배열로 받아 처리한다. | 서버에 파일 크기를 전송하고 지정한 크기만큼 파일을 전송한다. |
| | 프로토콜 송수신 | | 컨트롤서버와 송수신할 경우는 String으로, 스트리밍서버와 송수신할 경우는 byte 배열로 송수신. |

4. 장애요소 및 해결방법

[표 4] 장애요소 및 해결방법

| 장애요소 항목 | 장애요소 내용 | 해결방법 |
|-------------|---|---|
| OPEN JDK | OPEN JDK 11버전에서 VLCJ 프레임워크가 정상적으로 작동하지 않는 현상이 발생하였다. | 개발 환경을 VLCJ프레임워크가 작동 가능한 ORACLE JDK 8버전으로 변경하여 개발하였다. |
| MySQL 권한 문제 | Control Server와 Media Server에서 MySQL 데이터베이스에 접근할 때 접근 권한이 할당되지 않아 접근이 되지 않는 현상이 발생하였다. | 접근 권한을 주기 위해 각각의 서버를 관리자 권한으로 실행하였다. (sudo 명령어) |
| 네트워크 연결문제 | 서버가 Virtual Box의 Ubuntu 환경에서 동작하면서 가상환경의 네트워크 주소가 실제 네트워크 주소가 일치하지 않아 외부에서 접속이 안되는 문제가 발생하였다. | 컴퓨터의 방화벽을 해제하고, 실제 네트워크의 요청을 Virtual Box의 IP와 포트 번호에 매핑시키는 포트포워딩 설정을 통해 해당 문제를 해결하였다. |
| 동기화 | 멀티쓰레드 환경에서 다수의 사용자가 접속할 시 소켓을 스레드별로 할당하면서 한 소켓에 여러 스레드가 할당되는 동기화 문제가 발생하였다. | 쓰레드에 소켓을 할당하는 과정에서 Mutex를 사용하여 한 소켓을 가리키는 변수에 여러 스레드가 할당되는 현상을 해결하였다. |

5. 개발 후기

[표 5] 에로사항

| 에로사항 | 개발하면서 가장 어려웠던 점에 대한 극복방법 |
|-------------------------|--|
| RTSP | RTSP로 스트리밍 서버를 구현하고자 하였으나 응용 레벨 프로토콜이기 때문에 정형화된 코드를 사용하기 위해서 책 한권 분량의 코드를 이해하여야 본 프로젝트에 적용 할 수 있는 어려움이 있었다. 대신에 서버에서 TCP 프로토콜을 사용하여 영상 데이터를 전송하면 클라이언트에서 데이터를 받아서 바로 재생 가능한 progressive download 방식으로 구현하여 해결하였다. |
| JAVA Client C Server | JAVA로 구축한 Client와 C로 구축한 Server간의 프로토콜을 송수신하기 위해 문자열을 사용하였다. 기존에 JAVA의 데이터 저장 형태는 Big Endian으로 C의 Little Endian과 큰 차이가 있다. 따라서 클라이언트에서 정수형태의 데이터를 Byte배열로 송신할 시, 서버에서 정상적으로 읽지 못하였다. 하지만 문자열 데이터를 송수신하는 것은 아무런 제한사항이 없고, 데이터의 원본의 순서와 데이터가 그대로 유지된다는 점을 파악하여, 데이터를 송신할 때 '0'을 패딩으로 추가하여 클라이언트와 서버간의 통신을 할 수 있도록 하였다. |
| 전송 지연 현상 발생 | 클라이언트와 서버 간 프로토콜을 송신하고 수신하는 과정에서 프로토콜을 두 번 이상 송수신할 경우, 정상적으로 전달이 안 되고 버퍼에 남아있는 현상이 발생하였다. 따라서 두 번 이상 프로토콜을 송수신할 경우, 소켓을 강제로 닫고 여는 과정을 통해 해당 문제를 해결하였다. |

[표 6] 개발 후기

| 팀원 | 개발 후기 (소감) |
|-----|---|
| 계희준 | <p>이번 프로젝트를 통해 팀장의 역할이 중요하다는 것을 깨달았다. 팀장으로써 팀원 간 소통 하는 방법을 배우고 팀 단위의 계획을 세우는 방법, 각 팀원에게 역할을 배분하는 방법을 배울 수 있었다.</p> <p>네트워크 수업을 통해 직접 서버와 클라이언트를 만들어 봄으로써 서버-클라이언트 사이에 발생 할 수 있는 문제점과 이를 해결하는 방법을 배울 수 있었다. 또한 스트리밍 서버를 개발하는 과정에서 RTSP 프로토콜을 적용하지 못해 아쉬웠지만 다양한 스트리밍 기법을 접하고 또 소켓을 이용해 직접 구현해 볼 수 있어서 많은 것을 배우고 경험 할 수 있었다.</p> |
| 강지현 | <p>이번 학기 네트워크 프로그래밍 강의를 들으면서 막연하게 느껴졌던 클라이언트/서버 개념을 프로젝트를 통해 직접 익힐 수 있어 유익했다. 수업을 듣기 전에는 소켓 통신이나 메시지 프로토콜, C와 Java와 같이 서로 다른 언어로 소켓 통신을 할 경우 전달되는 메시지가 어떻게 되는지에 대해 알지 못했다. 하지만 프로젝트를 진행하면서, 직접 프로그램을 만들게 되면서 조금씩 이해 할 수 있었다.</p> <p>아쉬운 점으로는 MVC 패턴으로 설계를 해보려 했으나, 패턴에 대한 이해가 부족했던 것, 짧은 개발 시간으로 인해 java GUI 프로그래밍에 대해 충분한 사전 지식 없이 코드를 구성한 점이다. 특히 C언어로 작성된 두 서버와 통신할 때 메시지를 보내고 받는 것에서 어려움이 발생해, 결국 팀장을 비롯한 나머지 팀원들의 많은 도움을 받아서 완성하게 되었다.</p> |
| 박성진 | <p>이번 프로젝트를 통해 통신에 필요한 메시지 프로토콜에 대해 이해하였으며 직접 서버-클라이언트 간 메시지 프로토콜을 정의하여 통신 규약을 만들어 봄으로써 실제로 이러한 방식으로 통신이 됨을 이해하였다. 또한 DB 개발 역할을 맡아서 DB와 서버간의 통신에 대해 이해를 하는데 큰 도움이 되었습니다. 팀 프로젝트 경험이 많이 있진 않았는데 서로가 겹치는 부분에 대해 어떤식으로 데이터를 주고 받을지에 대해 토의하고 그에 맞게 프로그래밍을 하면서 팀 프로젝트에 대해 배웠습니다.</p> |
| 이상덕 | <p>기존에 NodeJS의 express프레임워크와 Firebase를 통해 서버를 구축한 경험이 있었다. 하지만 체계적으로 배운 것이 아니었기 때문에 체계적인 학습에 대한 갈증이 있었습니다. 이번 수업을 통해 네트워크의 구조에 대해서 제대로 학습하고, 로우 레벨의 소켓프로그래밍을 직접 경험함으로써 네트워크 프로그래밍에 대한 소양이 늘어난 것을 느꼈다.</p> |

JAVA로 VLC플레이어를 설치하면서 openJDK와 Oracle JDK간의 호환성 문제가 발생했었다. 이전에 Oracle에서 유료화를 했을때에는 아무런 실감이 나지 않았는데 이번에 해당 문제를 해결하면서 작지 않은 문제였다는 것을 느꼈다.

JAVA와 C간 통신과정에서 발생하는 문제를 문자열을 사용해서 해결하였다. C언어에서 자체적으로 String을 지원하지 않아 해당 문제를 해결하는데 큰 어려움을 겪었지만, C++와 Python과 같은 언어들이 String을 지원하는 것을 생각하면 다른 언어로 서버를 작성할 때 해당 기술을 적극적으로 활용할 수 있을것이라 생각하니 설렘을 느꼈다.

좋은 수업을 들어 정말 즐거웠다.