Open Source Software Project

: DGU Chat Final Report

Prof. Dongho Kim

과제	최종보고서
제출	2017년 06월 11일 일요일
번호	3조
팀장	컴퓨터공학과 2012111938 석정한
팀원	컴퓨터공학과 2013112043 심희오
	컴퓨터공학과 2013112048 강정석
	컴퓨터공학과 2014113128 천훼

1. 과제 목표

공개소프트웨어의 중요성과 사회 발전에 미치는 영향을 이해한다. 또한, 기존의 공개소프트웨어를 기반으로 새로운 공개소프트웨어를 개발하는 전 과정을 경험한다. 공개소프트웨어 개발환경에서 공개소스를 기반으로 팀 단위 협업프로그래밍 과제를 수행하여 유용한 소프트웨어를 제작하고 이를 공개한다.

특히, 3조는 WebRTC (Web Real-Time Communication) 기술을 이용한 화상 채팅 웹 어플리케이션 개발과, Qt 프레임워크을 이용하여 WebRTC를 지원하지 않는 브라우저에서도 동작할 수 있게 멀티 플랫폼, 하이브리드 어플레케이션 개발을 목표로 한다.

마지막으로 소프트웨어를 제작하고 개발 방법 및 사용법을 Github에 공개하여 사회에 기여한다.

2. 과제의 목적

프로그램을 개발할 때 필요한 지식과 기술들이 너무나 많고 개발 방법들이 매우 다양하다.

이 프로젝트의 예로 (화상 채팅 프로그램 개발 프로젝트) 서버 측에서는 클라이언트들의 데이터를 처리하는 소켓 프로그래밍 지식, 네트워크에 대한 이해, 네트워크 보안 등에 대해 알아야한다. 클라이언트 측에서는 해당 시스템 아키텍쳐와 운영체제를 고려하여 캠과 오디오를 접근하며 이를 서버측에 보내고 때론 서버 측에서 데이터를 받아 상대방의 모습을 화면에 출력해야한다. 특정 운영체제의 API를 사용해서 어플리케이션을 개발해도 되며, 다양한 GUI 프레임워크를 사용할수도 있다.

만약, 개발자가 서버와 클라이언트 모두 개발을 해야하거나, 서버나 클라이언트 프로그램들이 요구하는 지식의 일부를 가지고 있지 않을 때 어떻게 해결 할 수 있을까? 이 문제를 해결할 수 있는 방법은 바로 공개소프트웨어이다.

이번 과제는 공개소프트웨어를 이용해서 화상 채팅 어플리케이션을 개발한다. 이를 통해 얼마나 쉽고 빠르게 강력한 어플리케이션을 개발할 수 있는지 나타내어, 많은 이들이 공개소프트웨어의 필요성을 느끼게 하는 곳이 목적이다.

3. 사용된 기술들과 공개 소프트웨어

3조는 화상 채팅 프로그램을 개발하기 위해 WebRTC 기술과 Qt 프레임워크를 사용한다.

WebRTC 기술에 필요한 시그널링 서버 프로그램과 라이브러리를 공개 소프트웨어로 사용한다. 이 공개 소프트웨어는 Muaz Khan이 개발하고 Github에 공개한 RTCMultiConnection이다. (MIT 라이센스)

WebRTC를 지원하지 않는 브라우저에서도 동작할 수 있는 멀티 플랫폼, 하이브리드 어플레케이션 개발을 하기 위해 Qt 프레임워크 (LGPL 무료 라이센스)을 사용한다.

3.1 WebRTC (Web Real Time Communication)

WebRTC는 웹 브라우저 간에 플러그인 도움 없이 상호 통신할 수 있도록 설계된 JavaScript API이다. WebRTC는 구글이 오픈소스화한 프로젝트에서 기원하였으며, 그 뒤로 국제 인터넷 표준화 기구(IETF)가 프로토콜 표준화 작업을, W3C가 API 정의를 진행하였으며, 음성 통화, 영상 통화, P2P 파일 공유, 데이터 통신 등으로 활용 될 수 있다.

WebRTC를 지원하는 웹 브라우저는 크롬, 파이어폭스, 오페라, 엣지 등이 있다. 마이크로소프트사의 인터넷 익스플로러와 애플의 사파리는 지원을 하지 않는다.

WebRTC의 통신 방법은 클라이언트 Alice와 Bob이 시그널링 서버를 통해 Session Description protocol (SDP)을 주고 받고 서버를 거치지 않고 서로 P2P (Peer to Peer) 통신을 시작하게 된다.

P2P 연결이 시작되면 RTCPeerConnection이 클라이언트에 직접 연결하기 위한 시도가 이루어진다. 하지만 실제 네트워크상의 복잡성과 보안상 문제로 인해 P2P 통신을 위해서라면 STUN, TURN 서버를 이용해야한다.

STUN 서버는 외부 네트워크 주소를 얻는데 사용되며, TURN 서버는 직접(P2P) 연결이 실패할 경우 트래픽을 중계하는데 사용된다.

자세한 내용은 https://webrtc.org/ 에서 확인할 수 있다.

3.2 RTCMultiConnection

클라이언트의 화면에 나타내는 웹화면을 Front-End 영역, 클라이언트에서 전달된 데이터 및 서버관리를 Back-End 영역이라고 정의를 한다면,

WebRTC를 이용해 채팅 웹 어플리케이션을 개발 시 채팅 방을 입장하거나 사용자들의 화상 정보를 웹 브라이저에 출력하는 영역을 Front-End 영역, 클라이언트들의 SDP 정보를 교환하는 영역을 Back-End 영역이라고 볼 수 있다. 그래서 우리는 Javascript WebRTC API과 HTML5, CSS3를 이용해서 Front-End를 구성하고 다양한 언어와 Socket 프로그래밍으로 Back-End를 구성할 수 있다.

Front-End 영역은 HTML5, CSS3, Javascript 지식을 어느정도 소유하고있다면 WebRTC API 명세서를 확인해서 충분히 구현을 할 수 있다. Back-End 영역에서는 클라이언트가 1대1 화상채팅만을 한다면 보다 간단하게 구현을 할 수 있지만, 만약 여러 클라이언트 모두 화상채팅을 한다고 하면 이 문제는 절대 간단하게 구현할 수 없다. 수십, 수백, 수천명의 데이터를 처리하고 관리하는 이 문제는 프로그래머의 능력에 따라 달라진다.

우리는 위 문제를 해결 하기 위해 Github에 공개된 RTCMultiConnection를 이용해서 Front-End와 Back-End 영역을 구현했다. RTCMultiConnection은 Muaz Khan이 개발 하였고, MIT 라이센스를 가지고 배포하고 있다.

RTCMultiConnection은 WebRTC API를 한번더 가공해서 RTCMultiConnection Signaling Server. Back-End에 맞게 Front-End Javascript API를 제공한다.

RTCMultiConnecton Signaling Server는 Node.js를 이용해서 다대다 WebRTC 통신을 지원하게 설계되어 있다.

우리는 공개소프트웨어인 RTCMultiConnection을 이용해서 시그널링 서버를 구축하고,

Javascript API를 이용하여 다대다를 지원하는 화상채팅 웹 어플리케이션을 개발하였다.

Muaz Khan은 RTCMultiConnection Signaling Server 또한 미리 구축하여 공개하였으나, 우리는 Muaz Khan의 공개된 시그널링 서버를 사용하지 않고, RTCMulticonnection을 사용하여 Heroku(https://www.heroku.com/)클라우드에 직접 서버를 구축하였다. 구축한 시그널링 서버의 주소는 http://dguchat.herokuapp.com/ 이다.

Front-End 부분은 Github Page에 구축하여 시그널링 서버의 트레픽을 분산 시켰다. https://rudebono.github.io/dguchat/ 에서 서비스를 이용할 수 있으며, 자세한 코드는 https://github.com/rudebono/dguchat/tree/gh-pages 에서 확인 가능하다.

3.3 Qt

Qt컴퍼니에서 개발한 GUI 프로그램 개발에 널리 쓰이는 크로스 플랫폼 프레임 워크이다. GUI 뿐만 아니라 콘솔 프로그램과 같은 비GUI 프로그램 개발에도 사용된다. 주로 C++을 기반으로 하고 있지만, 파이썬, 루비, C, perl, 파스칼과도 연동이 가능하다. 또한 Qt는 "write once, complie anywhere"을 표방하여 하나의 코드로도 다양한 플랫폼에서 빌드가 가능하도록 지원하고 있다. (유료, LGPL, GPL)

WebRTC를 이용해서 웹 어플리케이션을 개발 한다면, 클라이언트들은 필히 WebRTC를 지원하는 브라우저를 이용하여 서비스를 이용해야한다. 하지만 클아이언트들이 WebRTC를 지원하지 않는 브라우저가 없거나, 새로 브라우저를 설치를 하는 것에 대해 반대를 한다면 이 문제는 발생한다. 그렇다면 우리는 클라이언트들의 플랫폼 환경에서 추가 설치되는 브라우저 없이 WebRTC를 지원하는 어플리케이션을 개발하려면 어떻게 해야할까? 새로운 브라우저를 만들 되 필요한 기능만 남겨놓고 (WebRTC 웹 어플리케이션에 접근하고 서비스하는 것.) 다른 기능들을 빼버리면 된다. 하지만 브라우저를 개발한다는 것은 그리 쉬운일이 아니다. 네트워크, 언어처리, 오디오, 그래픽 처리 등 해결해야 문제들이 너무나 많다. 그래서 우리는 브라우저를 개발하기 위해 웹 브라우저 개발 프레임워크를 사용한다. 이 또한 공개 소프트웨어 이다.

공개된 브라우저 개발 프레임워크는 대표적으로 WebKit, Chromium이 있다. 이 프레임워크와 다양한 언어를 합하고 GUI API를 사용해서 클라이언트 플랫폼에 맞는 프로그램을 개발 할 수 있다. 하지만 클라이언트들의 플랫폼은 다양하기 때문에 여러 플랫폼을 하나하나씩 개발하는데 큰 문제가 있다. 그래서 크로스 멀티 플랫폼 프레임 워크인 Qt를 사용한다.

Qt는 qtWebkit, qtWebengine이란 이름으로 WebKit과 Chromium를 동시에 지원하며, 우리는 qtWebengine를 사용하여 하이브리드 어플리케이션을 개발하였다.

qt에 대한 자세한 내용은 https://www.qt.io/ , WebKit에 대한 자세한 내용은 https://webkit.org/ , Chromium에 대한 제세한 내용은 https://www.chromium.org/ 에서 확인 가능하다.

4. 결과물 공개

모든 소스코드, 문서를 Github (https://github.com/rudebono/dguchat)에 모두 공유하였다.

이 과제에서 프로그램 구현 부분은 총 3가지로 나눌 수 있다. RTCMultiConnection를 이용한 시그널링 서버, RTCMultiConnection API를 이용한 Front-End, Qt를 이용한 하이브리드 어플리케이션이다.

시그널링 서버 (Back-End)에 대한 소스코드와 자세한 내용은 아래의 주소에서 확인할 수 있다. https://github.com/rudebono/dguchat/tree/master/server

화상 채팅 웹페에지 (Front-End)에 대한 소스코드와 제세한 내용은 아래의 주소에서 확인할 수 있다. https://github.com/rudebono/dguchat/tree/gh-pages

Qt를 이용한 하이브리드 어플리케이션에 대한 소스코드와 자세한 내용은 아래의 주소에서 확인 할 수 있다. https://github.com/rudebono/dguchat/tree/master/client

과제 진행을 하면서 제출했던 모든 문서들은 아래 주소에서 확인 가능하다. https://github.com/rudebono/dguchat/tree/master/homework

5. 결과물

https://rudebono.github.io/dguchat/

WebRTC를 이용한 동국대학교 화상 채팅 어플리케이션