

WebRTC 를 이용한 P2P 파일 공유 웹 애플리케이션 설계 및 구현

김진우*, 박상원*

*한국외국어대학교 정보통신공학과

e-mail : kjw749@naver.com, swpark@hufs.ac.kr

Design and Implementation of a Web Application for P2P file sharing on WebRTC

Jin-Woo Kim*, Sang-Won Park*

*Dept. of Information and Communications Engineering, Han-Kook University of Foreign Studies

요 약

스마트기기 간 파일을 공유할 경우, 사용자는 파일 공유 프로그램을 설치해 이를 이용하거나 외부저장장치를 이용해 파일을 공유한다. 클라우드 저장소에 파일을 저장해 이를 공유하는 웹 애플리케이션을 사용할 경우, 클라우드 저장소의 제한된 크기로 인해 파일의 크기가 제한되는 경우가 있다. 본 논문에서는 기존 파일 공유 방법의 단점을 해결하기 위해 P2P 파일 공유 웹 애플리케이션을 제시한다. P2P 파일 공유 웹 애플리케이션을 이용하면 기존에 설치된 브라우저만을 이용해 파일 용량 제한 없는 파일 공유가 가능하다. HTML5 표준의 WebRTC 를 이용하면 브라우저만을 이용해 연결지향 양방향 P2P 통신이 가능하다. 본 논문에서는 P2P 파일 공유 웹 애플리케이션 구현에 앞서 P2P 파일 공유 프로토콜을 제시한다. 본 논문에서 제시하는 P2P 파일 공유 프로토콜은 JSON 메시지와 메시지 핸들러를 이용한 브라우저 간 비동기적 RPC(Remote Procedure Call) 형태로 설계되었다. 본 논문에서 설계한 프로토콜을 이용해 P2P 파일 공유 웹 애플리케이션을 구현하였다.

1. 서론

기존에 실생활에서 사용하는 스마트기기 간 파일 공유 방법에는 다음과 같은 단점들이 존재한다. 웹 애플리케이션이 아닌 DropBox 와 같은 응용프로그램을 이용해 파일을 공유할 경우, 사용자는 응용프로그램을 설치해야 한다. USB 와 같은 외부 저장장치를 이용해 스마트기기 간 파일을 공유할 경우, 사용자는 파일 공유를 위해 장비를 준비해야 한다. 이메일과 같은 클라우드 웹 애플리케이션을 이용해 파일을 공유하는 경우, 클라우드 저장장치의 용량으로 인해 공유하고자 하는 파일의 용량이 제한되는 경우도 있다. 본 논문은 기존의 파일 공유 방법들의 단점을 해결하는 방안으로 P2P 파일 공유 웹 애플리케이션을 제안한다. 사용자는 기존에 설치된 웹 브라우저를 이용해 클라우드 저장장치의 용량에 상관없이 다른 스마트기기와의 P2P 파일 공유가 가능하다.

본 논문에서는 P2P 파일 공유 웹 애플리케이션 구현에 앞서 WebRTC[1][2]를 이용한 P2P 파일 공유 프로토콜을 설계한다. HTML5 표준의 WebRTC 를 이용하면 브라우저만을 이용해 연결지향 양방향 P2P 통신이 가능하다. 본 논문에서 제시하는 P2P 파일 공유 프로토콜은 JSON 메시지와 메시지 핸들러를 이용한 비동기적 RPC(Remote Procedure Call) 형태로 설계되었다. 본 논문에서 설계한 프로토콜을 이용해 P2P

파일 공유 웹 애플리케이션을 구현할 수 있다.

2. 관련 연구

2.1 브라우저 간 데이터 통신 API

P2P 파일 공유 웹 애플리케이션을 개발하기 위해서는 브라우저를 이용해 데이터 통신을 할 수 있어야 한다. 본 절에서는 브라우저를 이용해 데이터 통신을 하는 API 들을 비교하고 각각의 API 가 P2P 파일 공유에 적합한지 파악한다.

API	프로토콜	연결지향	서버에 파일을 업로드하는 과정	신뢰도
HTTP	HTTP	X	필요	O
WebSocket	TCP	O	필요	O
WebRTC	SCTP	O	불필요	O(설정 가능)

〈표 1〉 브라우저 간 데이터통신 API

HTML5 표준 이전에 브라우저는 HTTP 프로토콜을 통해서만 데이터 통신이 가능했다. 표 1 에서 보는 바와 같이 HTTP 는 기본적으로 비연결지향 통신이기 때문에 데이터를 보낼 때마다 핸드셰이크를 해야 하는 오버헤드가 있다. 용량이 큰 파일을 전송할 경우, 브라우저는 데이터를 여러 번에 걸쳐 전송하게 되는

데 전송할 때마다 핸드셰이크가 발생하기 때문에 파일 전송이 오래 걸린다.

표 1 에서 보는 바와 같이 HTML5 표준의 WebSocket[3]을 이용하면 브라우저는 TCP 프로토콜 기반 소켓 통신이 가능하다. WebSocket 은 서버와 클라이언트 간에 연결지향 양방향 통신을 제공해 한 번의 핸드셰이크 만으로 데이터를 지속해서 전송할 수 있다. 일반적으로 WebSocket 을 이용해 파일 공유 웹 애플리케이션을 제작할 경우, 파일은 서버를 거쳐 원격지 브라우저에 도착하게 된다. 로컬 브라우저는 원격지 브라우저의 파일이 서버에 업로드가 완료된 후에 파일을 다운로드할 수 있기 때문에 서버에 파일 업로드가 완료될 때까지 기다려야 한다[4]. 일반적으로 아니지만 WebSocket 을 이용해 P2P 통신을 구현할 경우, 코드구현에 어려움이 있다. WebSocket 코드는 WebSocket 서버에 사용되는 코드와 브라우저에서 클라이언트에 사용되는 코드가 다른 형태로 나뉘어 있다. 서버는 클라이언트의 연결을 기다리는 역할을 하고 클라이언트는 서버로 연결을 요청하는 역할을 한다. WebSocket 으로 P2P 통신을 구현할 경우 로컬브라우저는 원격지 브라우저의 연결을 기다리는 동시에 원격지 브라우저에 연결요청을 할 수 있어야 한다. 기존의 각각 사용하는 코드로 이를 구현하는 것을 복잡하다. 또한, 원격지 브라우저가 NAT (Network Address Translation)를 사용하는 네트워크에 존재할 경우, 로컬 브라우저는 원격지 브라우저의 IP 주소와 포트 번호를 알지 못해 P2P 통신을 연결하기 어렵다.

표 1 에서 보는 바와 같이 HTML5 표준의 WebRTC 를 이용하면 브라우저 P2P 통신이 가능하다. WebRTC 의 P2P 연결 API 를 이용하면 브라우저는 NAT 네트워크 내 브라우저와 P2P 통신할 수 있다. 이를 통해 WebRTC 를 이용하면 브라우저 간 서버에 파일을 업로드 하지 않고 파일을 공유할 수 있다. WebRTC 는 SCTP 프로토콜을 사용한다. SCTP 는 기본적으로 실시간 비디오, 오디오 통신과 같이 빠른 데이터 전송을 위해 UDP 처럼 신뢰성 없는 메시지 기반 통신을 사용한다. 하지만 데이터 손실 없는 P2P 데이터 통신을 지원하기 위해 신뢰성 있는 통신으로도 설정할 수 있다. WebRTC 의 신뢰성 있는 P2P 데이터 통신 API 를 사용하면 P2P 파일 공유 웹 애플리케이션을 구현할 수 있다.

2.2 브라우저 간 P2P 파일 공유 웹 애플리케이션

이름	파일 전송	파일 요청	로그인 방법
ShareDrop ¹⁾	O	X	URL
Sharfest ²⁾	O	X	URL
FilePizza ³⁾	O	X	URL
Justbeamit ⁴⁾	O	X	URL

〈표 2〉 브라우저 간 P2P 파일 공유 웹 애플리케이션

표 3 에서 보는 바와 같이 P2P 파일 공유 웹 애플리케이션을 이용해 브라우저 간 파일을 공유할 수 있다. 표 3 에서의 파일 전송은 파일은 원격지 브라우저

로 파일을 전송하는 기능을 뜻하고 파일 요청은 원격지 브라우저의 특정 파일을 선택해 다운로드하는 기능을 뜻한다. 표 3 에서 보이는 바와 같이 모든 P2P 파일 공유 웹 애플리케이션들은 파일 전송 기능을 가지고 있다. 공유하고자 하는 파일을 로컬 브라우저에서 업로드하면 URL 이 반환되고 이 URL 을 이용해 원격지 브라우저에서 접근하게 되면 로컬 브라우저에서 업로드한 파일이 자동으로 다운로드되는 기능을 제공한다. 웹의 URL 을 이용해 원격지 브라우저와 연결하게 되면 ID 를 생성해서 로그인하지 않아도 쉽게 파일을 공유할 수 있다.

표 3 에서 보이는 바와 같이 표 3 의 모든 웹 애플리케이션들은 로컬 브라우저에서 원격지 브라우저로 파일을 일방적으로 전송한다. 하지만 일방적인 파일 전송이 아닌 원격지 브라우저로 원하는 파일의 전송을 요청하는 파일 요청 기능을 제공하고 있지 않다. 일방적인 파일 전송만 있으면 원격지 브라우저는 로컬 브라우저가 전송하는 파일 이외에 자신이 원하는 파일을 전송받을 수 없다. 본 논문에서는 양쪽 브라우저 간에 활발한 파일 공유를 위해 파일 전송 이외에도 파일 요청을 위한 대한 프로토콜을 설계하였다.

3. P2P 파일 공유 프로토콜

본 절에서는 P2P 파일 공유 웹 애플리케이션에서 사용되는 P2P 파일 공유 프로토콜에 대해 설명한다.

프로토콜	실행 함수	기능
파일 전송	sendFile	원격지 브라우저로 파일 전송
파일 요청	download	원격지 브라우저의 파일을 선택해 다운로드
파일 리스트 요청	getFileList	원격지 브라우저의 파일 리스트 확인

〈표 3〉 P2P 파일 공유 프로토콜

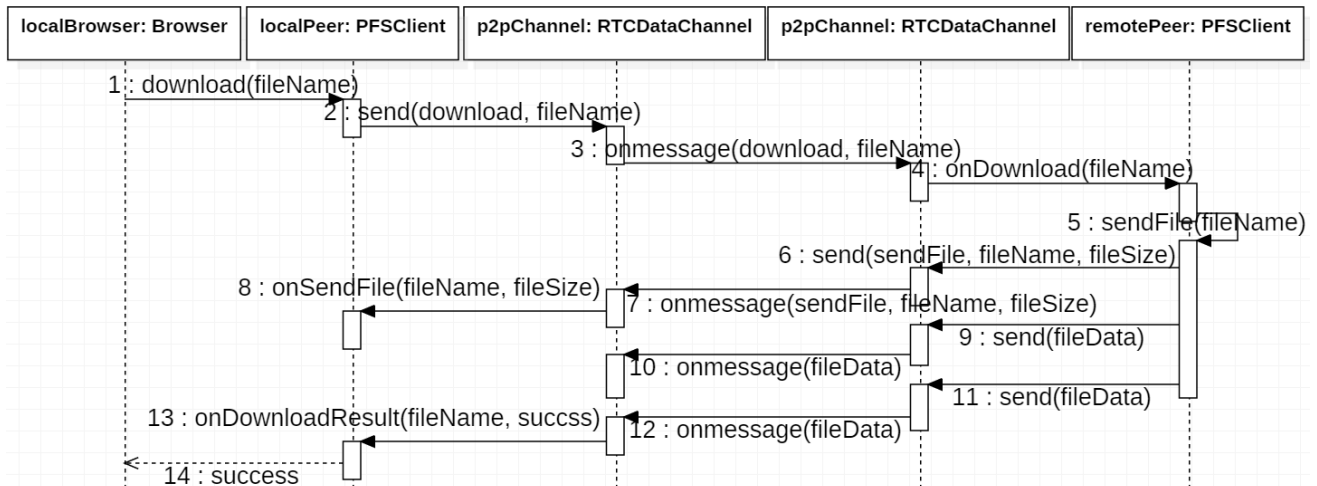
본 논문에서는 표 3 에서와같이 파일 공유를 위해 파일 전송, 파일 요청, 파일 리스트 요청 프로토콜을 정의하였다. 첫 번째, 파일 전송 프로토콜의 경우 로컬 브라우저에서 원격지 브라우저로 파일을 전송하는 프로토콜이다. 본 논문에서는 sendFile 함수 실행 시 수행된다. 두 번째, 파일 요청 프로토콜의 경우 로컬 브라우저에서 원격지 브라우저가 제공하는 특정 파일을 선택해 다운로드 받는 프로토콜이다. 파일 요청 프로토콜은 본 논문에서는 download 함수 실행 시 수행된다. 세 번째, 파일 리스트 요청 프로토콜은 원격지 브라우저가 제공하는 파일의 리스트를 확인하는 프로토콜이다. 파일 리스트 요청 프로토콜은 본 논문에서는 getFileList 함수 실행 시 수행된다.

1) ShareDrop: <https://www.sharedrop.io/>

2) Sharfest: <https://www.sharfest.me/>

3) FilePizza: <https://file.pizza/>

4) JustBeamIt: <https://www.justbeamit.com/>



(그림 1) 비동기적 RPC 동작 과정 예시 (파일 요청 프로토콜)

3.1 비동기적 RPC(Remote Procedure Call)

로컬 브라우저와 원격지 브라우저 간에 통신에는 주로 비동기적 통신이 사용된다. 비동기적 통신은 메시지 기반 통신으로 통신 장비 간에 특정 내용이 담긴 메시지를 전달하는 방식으로 통신한다. 원격지에서는 상대방으로부터 전달받은 메시지의 내용에 따라 해당 메시지를 처리하는 함수를 호출해 전달받은 내용에 따른 동작을 수행한다.

본 논문에서는 비동기적 통신에 사용되는 메시지와 메시지 처리 함수를 비동기적 RPC 형태의 함수와 콜백 함수로 정의하였다. 기존에 원격지 PC의 파일시스템의 파일 데이터를 가져와 사용하는 NFS 프로토콜[5]은 동기적 RPC를 사용하는 것과 다르게 본 논문은 GUI와 비동적으로 동작하는 자바스크립트의 특성에 맞춰 RPC를 비동기적으로 사용하였다. 웹과 같은 GUI 프로그램은 이벤트 중심의 비동기적 함수 호출 형태로 구성된다. GUI에서 이벤트가 발생하면 이를 처리하기 위해 웹 애플리케이션은 이벤트의 내용에 따라 이벤트의 내용이 담긴 메시지를 이를 처리하는 자바스크립트 함수로 전달한다. 메시지를 처리하는 함수는 이 메시지를 처리하기 위해 메시지에 명시된 이벤트의 내용에 따라 콜백 함수를 호출한다. 본 논문에서 제안하는 방법도 이와 유사하다. 본 논문의 비동기적 RPC는 함수 호출과 결과 반환의 2단계로 실행된다. 첫 번째, 함수 호출 단계에서는 로컬 브라우저에서 특정 함수를 호출하면 JSON 메시지를 통해 원격지 브라우저에 해당 함수에 호출을 알린다. 원격지 브라우저에서는 JSON 메시지에 정의된 함수명에 따라 해당 함수의 콜백 함수를 호출한다. 두 번째 결과 반환 과정은 원격지 브라우저는 콜백 함수의 결과를 JSON 메시지 형태로 로컬 브라우저에 전달된다. 로컬 브라우저는 원격지 브라우저에서 보낸 메시지의 콜백 함수명에 따라 최종적으로 함수의 결과를 처리하는 콜백 함수를 호출한다.

그림 1에서는 비동기적 RPC 형태의 프로토콜의 예시로 파일 요청 프로토콜을 사용하였다. 로컬 브라우

저는 download 함수의 실행 시 download 함수의 실행을 알리는 내용의 JSON 메시지를 WebRTC API의 RTCDataChannel를 통해 원격지 브라우저로 전송한다. 원격지 브라우저는 메시지의 적힌 함수명에 따라 콜백 함수를 호출한다. 메시지에 적힌 함수명 download에 따라 콜백 함수 onDownload가 호출이 되고 메시지에 저장된 매개변수는 콜백 함수 onDownload의 매개변수로 사용된다. onDownload 함수를 통해 원격지 브라우저는 파일 저장에 필요한 파일의 이름과 사이즈를 전송하고 파일 데이터를 SCTP 프로토콜 메시지의 크기인 16KB 단위로 나눠서 로컬 브라우저로 전송한다. 최종적으로 파일 데이터가 파일의 크기만큼 모두 전송되면 로컬 브라우저는 onDownload의 콜백 함수인 onDownloadResult를 호출해 download 함수의 결과를 처리한다.

3.2 메시지 핸들러

msgHandler : Javascript Object

funcName (=key)	Call back function (=value)
download	onDownload
onDownload	onDownloadResult
getFileList	onGetFileList
onGetFileList	onGetFileListResult

p2pMsgHandler["onRead"] = onReadResult

(그림 2) 메시지 핸들러

로컬 브라우저에서 수행한 함수에 따라 원격지 브라우저는 콜백 함수를 호출한다. 이를 위해 본 논문에서는 그림 2에서 보이는 바와 같이 함수 및 콜백 함수를 메시지 핸들러 객체에 저장해 사용하였다. 자바스크립트의 객체는 연관 배열 (Associative Array) 형태로 되어 있다. 이를 이용해 메시지 핸들러로 명칭이 된 자바스크립트 객체에 함수명을 키 값으로, 콜

백 함수를 값으로 저장해 사용한다. 로컬 브라우저가 함수를 실행하면 함수명과 매개변수가 저장된 JSON 메시지가 원격지 브라우저에 전달되면 메시지 핸들러는 메시지에 정의된 함수명을 키 값으로 해 해당 함수의 콜백 함수를 메시지에 정의된 매개변수와 함께 실행한다.

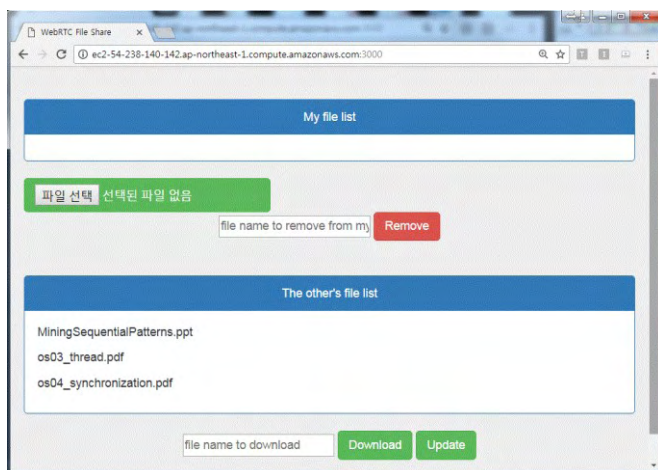
로컬 브라우저 함수	원격지 브라우저 함수
getFileList	onGetFileList
onGetFileListResult	
sendFile	onSendFile
onSendFileResult	
download	onDownload
onDownloadResult	

<표 4> p2pMsgHandler

표 4 과 같이 본 논문에서는 p2pMsgHandler 로 명칭된 메시지 핸들러를 사용하였다. 각각 프로토콜에서 로컬 브라우저에 실행하는 함수, 원격지 브라우저에서 실행할 콜백 함수, 원격지 브라우저에서 실행한 콜백함수의 결과를 최종적으로 로컬 브라우저에서 처리하는 콜백 함수를 저장해 사용하였다.

4. 실험 및 결과 분석

P2P 파일 공유 프로토콜을 사용해 실제 P2P 파일 공유 웹 애플리케이션 구현해 보았다. 그림 3 의 P2P 파일 공유 웹 애플리케이션은 P2P 파일 공유 프로토콜을 기반으로 로컬 브라우저에서 원격지 브라우저가 업로드한 파일의 리스트를 확인 후 원격지 브라우저에서 제공하는 특정 파일을 선택해 다운받도록 제작하였다. 이를 통해 P2P 파일 공유 프로토콜을 통해 P2P 파일 공유 웹 애플리케이션 구현이 가능함을 입증할 수 있었다.



(그림 3) P2P 파일 공유 웹 애플리케이션

5. 결론 및 향후 계획

본 논문에서는 WebRTC API 를 이용해 P2P 파일 공유 웹 애플리케이션을 설계 및 개발하였다. 메시지 핸들러를 이용한 비동기적 RPC 형태의 프로토콜을 통해 P2P 파일 공유 웹 애플리케이션을 개발할 수 있었다. 추후에 본 논문에서 설계한 P2P 파일 공유 프로토콜을 일반화해 라이브러리로 구현할 계획이다.

참고문헌

- [1] Rob Manson, Getting Started with WebRTC, UK: Packt Publishing, 2013.
- [2] Sam Dutton "Getting Started with WebRTC", <https://www.html5rocks.com/en/tutorials/webrtc/basics/#rtcdatachannel>, 201
- [3] Malte Ubl , Eiji Kitamura, "Introducing WebSockets: Bringing Sockets to the Web", <https://www.html5rocks.com/en/tutorials/websockets/basics/>, 2010
- [4] Quanfeng Duan, Zhenghe Liang and Limin Wang, "Efficient File Sharing Scheme Based on WebRTC", 3rd International Conference on Material, Mechanical and Manufacturing Engineering, p730-734, 2015.
- [5] R. Sandberg, D. Goldberg, S. Kleiman, D. Walsh, and B. Lyon, "Design and implementation of the Sun Network", In Proc. of USENIX Summer Technical Conf., June 1985